VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV STROJÍRENSKÉ TECHNOLOGIE
INSTITUTE OF MANUFACTURING TECHNOLOGY

VYUŽITÍ PARAMETRICKÉHO PROGRAMOVÁNÍ PRO OBRÁBĚNÍ OBECNÝCH PLOCH
THE USE OF PARAMETRICAL PROGRAMMING FOR COMPLEX PART MACHINING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Petr Mohyla**
AUTHOR

VEDOUCÍ PRÁCE — **prof. Ing. Miroslav Píška, CSc.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav strojírenské technologie
Student:	**Bc. Petr Mohyla**
Studijní program:	Strojní inženýrství
Studijní obor:	Strojírenská technologie
Vedoucí práce:	**prof. Ing. Miroslav Píška, CSc.**
Akademický rok:	2016/17

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Využití parametrického programování pro obrábění obecných ploch

Stručná charakteristika problematiky úkolu:

Využití parametrického programování pro obrábění obecných ploch s podporou uživatelských cyklů pro řídicí systémy Sinumerik.

Cíle diplomové práce:

1. Teoretický rozbor problému.
2. Navržené varianty řešení - matematické, technologické.
3. Experimentální zkoušky.
4. Diskuze výsledků.

Seznam literatury:

LYNCH, M. Parametric programming for computer numerical control machine tools and touch probes: CNC's best-kept secret. Dearborn: Society of Manufacturing Engineers, 1997, 433 s. ISBN 0872634817.

KRUTH, J.P., LAUWERS, B., DOTREMONT, J., DEJONGHE, P. Optimised NC-toolpath generation for 5-axis machining of complex surfaces. Machining impossible shapes. Kluwer Academic Publishers, 1999, pp. 343-350, ISBN 0-412-84680-2.

MERDOL, D.S ., ALTINTAS, Y. Virtual Simulation and Optimization of Milling Operations: Part I – Process Simulation, Trans. ASME, J. Manufac. Sc. and Eng., 2008, vol. 130, pp. 051004.

MATTSON, M. CNC programming: principles and applications. Albany: Delmar, 2002, 358 s. ISBN 0766818888.

MCMAHON, Ch., BROWNE, J. CADCAM: principles, practise and manufacturing management. 2nd ed. Harlow: Prentice Hall, 1998, 665 s. ISBN 0201178192.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2016/17.

V Brně, dne 6. 11. 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA STROJNÍHO INŽENÝRSTVÍ DĚKANÁT

L. S.

prof. Ing. Miroslav Píška, CSc.
ředitel ústavu

doc. Ing. Jaroslav Katolický, Ph.D.
děkan fakulty

ABSTRAKT

Tato diplomová práce je zaměřená na parametrické programování a tvorbu uživatelských cyklů u CNC obráběcích strojů v řídicím systému Sinumerik 840 D sl / 828 D. První část obsahuje teoretický úvod do programování a tvorby grafické podpory cyklu. V druhé části jsou uvedeny praktické ukázky s popisy a vysvětlivkami, které mohou fungovat jako návod pro tvorbu parametrických programů. Tyto ukázky jsou v závěru práce experimentálně ověřeny.

Klíčová slova

parametrické programování, Sinumerik, uživatelské cykly

ABSTRACT

This diploma thesis is focused on parametric programming and creation of user cycles in CNC machine tools in control system Sinumerik 840 D sl / 828 D**.** The first part contains a theoretical introduction to programming and creation of graphical support of the cycle. In the second part there are practical examples with descriptions and explanations that can act as a guide for creation of parametrical programs. These samples are experimentally verified at the end of the work.

Keywords

parametrical programming, Sinumerik, user cycles

BIBLIOGRAFICKÁ CITACE

MOHYLA, P. *Využití parametrického programování pro obrábění obecných ploch.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2017. 62 s. Vedoucí diplomové práce prof. Ing. Miroslav Píška, CSc.

PROHLÁŠENÍ

Prohlašuji, že jsem diplomovou práci na téma **Využití parametrického programování pro obrábění obecných ploch** vypracoval samostatně s použitím odborné literatury a pramenů, uvedených na seznamu, který tvoří přílohu této práce.

---------- ----------

Datum Bc. Petr Mohyla

PODĚKOVÁNÍ

Děkuji tímto panu prof. Ing. Miroslavu Píškovi, CSc. za cenné připomínky a rady, které mi při vypracování diplomové práce věnoval. Dále pak děkuji Ing. Aleši Polzerovi, Ph.D. a Ing. Jiřímu Urbanovi za konzultace k použití systému Sinumerik a panu Jiřímu Čechovi s Janem Pokorným za jejich čas a spolupráci při experimentech.

OBSAH

ÚVOD

Konvenční stroje jsou dnes už téměř vytlačeny a nahrazeny moderními CNC *(Computer Numerical Control)* stroji s řídicím systémem. Je tomu tak díky tomu, že CNC obráběcí stroje dosahují výborných parametrů v obrábění jako je např. produktivita, kvalita povrchu a schopnost obrábět tvarově složité součásti, na které bylo dříve zapotřebí jednoúčelové stroje.

Příprava práce na CNC strojích se v průběhu let hodně změnila. Od dob, kdy se používaly děrné štítky nebo pásky, se v této oblasti odehrál velký pokrok. Nástroje, které usnadňují programování a jsou vyučovány jako základy, nebyly vždy samozřejmostí. Patří mezi ně např. kompenzace rádiusu nástroje nebo programové cykly, díky kterým lze rychle a snadno soustružit složitější tvary nebo vyrobit složitější závity. Velmi užitečným nástrojem pro zvýšení efektivity práce na CNC strojích je parametrické programování. Bohužel tento nástroj nebyl úplně přijat a relativně málo programátorů jej dokáže využívat, ačkoliv parametrické programování bylo představeno už před třiceti lety [1].

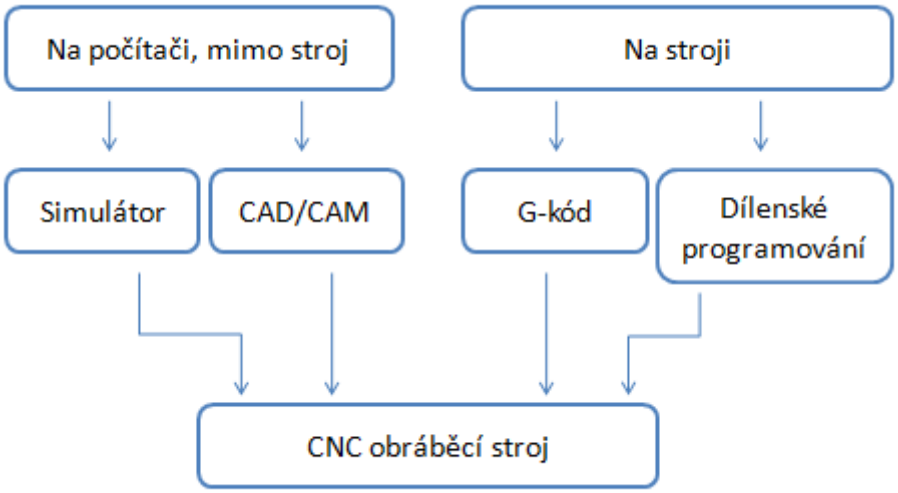
Jedním z cílů této práce je seznámit čtenáře s možnostmi parametrického programování a ukázat, jak se tento typ programování dá efektivně využít. První kapitola je teoretická. Jsou v ní shrnuty základy programování dle ISO normy a dále je popsáno parametrické programování v řídicím systému Sinumerik. Druhá kapitola je věnována obrábění korozivzdorných ocelí, protože jeden z vytvořených cyklů je vytvořen právě k aplikaci na tento typ ocelí. Ve třetí kapitole je popsán návrh řešení dvou programových cyklů, sestavených na základě parametrického programování a ve čtvrté kapitole jsou programy experimentálně ověřeny a popsány jejich důležité části.

1 TEORIE K PROBLEMATICE PROGRAMOVÁNÍ

V programu pro jakýkoli výrobek jsou obsaženy geometrické a technologické informace, které musí programátor znát, aby napsal vhodný kód pro vyrobení dané součásti. Technolog (programátor) musí dobře znát metodiku a možné způsoby programování, aby zvolil vhodnou, nejefektivnější cestu tvorby programu a výroby součásti, která je s tím úzce spjata.

1.1 Metodika programování

První rozhodnutí, které musí programátor udělat je, kde se bude program utvářet. Přímo na stroji, ručním programování G-kódu nebo využitím dílenského programování jako je např. ShopTurn, ShopMill. Druhou variantou je programování mimo stroj s využitím simulátoru nebo vygenerováním kódu za pomoci CAD/CAM softwaru a následného post-processingu, přičemž stroj může během přípravy programu pracovat, což značně zvyšuje efektivitu výroby. Na obr. 1.1 je základní rozdělení metod tvorby programu [3, 4].



Obr. 1.1 Základní rozdělení metod tvorby NC programu.

1.1.1 ISO programování

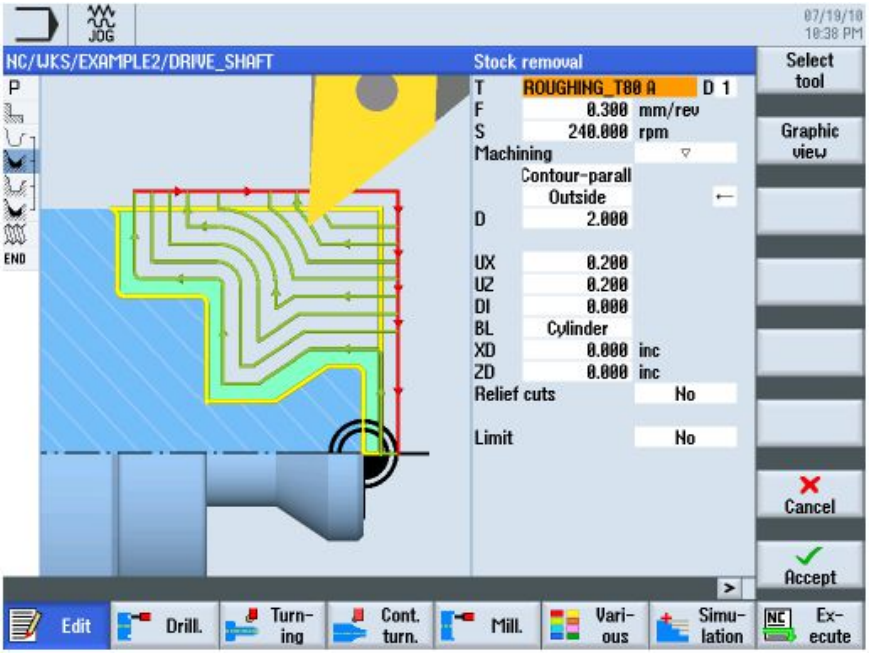
Programování dle normy ISO je základní a nejběžnější forma programování. Je standardizované normou a je využíváno všemi velkými výrobci řídicích systémů (dále jen ŘS) jako je např. Fanuc, Heidenhain, Siemens aj. Některé z kódů (např. G-kód, M-kód) nejsou standardní a jsou obsazovány výrobci ŘS dle zaměření stroje. Program je tvořen z tzv. **bloků** (řádků), které mohou být očíslovány pro větší přehlednost. V každém bloku může být několik funkcí jako např. G, F, M aj., které určují trajektorii, polohu, otáčky a další parametry. Výhodou programování dle normy ISO je totožnost a jednoznačnost na všech ŘS. Nevýhodou, vůči ostatním metodám, však mnohdy může být zdlouhavé programování a větší riziko výskytu lidské chyby [5, 6].

Příklad ručního programování dle normy ISO vypadá takto:

```
N40 T3 H3 D1          ; výměna nástroje
N50 G97 S300 M4       ; konstantní řezná rychlost, otáčky
N60 G0 X100 Z10       ; přejezd do výměny nástroje
N70 G0 X55 Z2         ; přejezd před obrobek
N80 G1 X60 Z-30       ; pracovní posuv po přímce
N90 G0 X60 Z2
```

1.1.2 Dílenské programování

U dílenského programování není zapotřebí znalosti programovacího jazyka. Namísto programování, obsluha stroje sestavuje pracovní postup a jednoduchými operacemi vytváří NC program. Díky přehledné struktuře programu se dají případné změny rychle editovat. Pro jednoduché součásti je tento způsob programování vysoce efektivní a velmi často využívaný, avšak obsluha musí být kvalifikovaná. Nevýhodou je, že prostředí dílenského programování si utváří každý výrobce ŘS sám, tudíž mají odlišná ovládání. Tento způsob však není vhodný pro pokročilejší programování. Na obr. 1.2 je ukázka grafického rozhraní pro dílenské programování v systému ShopTurn od firmy Siemens [8].

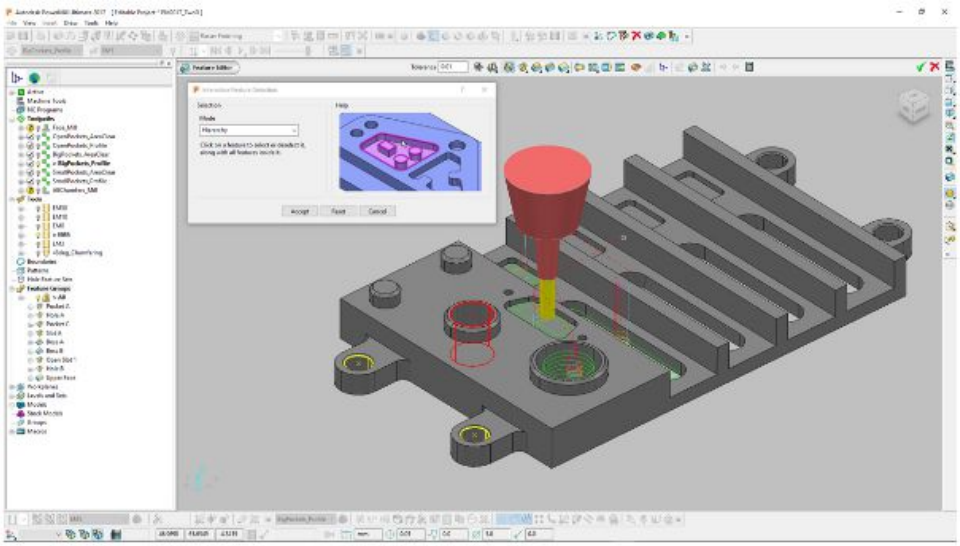


Obr. 1.2 Ukázka grafického rozhraní pro dílenské programování v systému ShopTurn [8].

1.1.3 Využití CAD/CAM software

CAD/CAM technologie se stále více využívá v praxi. Zpravidla zajišťuje kvalitu obrobené plochy, snižování výrobních časů a délku přípravy programu. Některé typy výrobků ani nelze naprogramovat ručně, nebo jen velmi pracně. Dobrým příkladem jsou formy na lisování do automobilového průmyslu, které využívají tvarově složité 3D plochy. Ekonomický přínos pro podnik, vybavený takovýmto softwarem, je značný, avšak je třeba brát na zřetel počáteční náklady za software a případné průběžné výdaje za aktualizace [5].

Pro práci s CAD/CAM musí mít programátor nejdříve 2D nebo 3D model součásti. Následuje práce s CAM, tvorba technologie - zadávání strategií, nástrojů apod. Dalším krokem je převedení strategie pomocí vhodného post-procesoru do ISO kódu. Často se může stát, že se v programu důsledkem špatně provedeného post-procesingu vyskytne chyba. Programátor musí znát základní ISO programování, aby mohl chyby odstranit [4].



Obr. 1.3 Ukázka pracovního prostředí v softwaru PowerMill od firmy Delcam [9].

1.1.4 Parametrické programování

V případě, že je zapotřebí obrábět výrobky, které jsou limitované svým tvarem a nebere se v úvahu efektivita jak tvorby programu, tak následného obrábění, programátor si vystačí s běžným programováním v G-kódu. Ale takovéto programování neumožňuje použití proměnných, funkcí, logických operací, ani smyček, což dělá konvenční programování velmi omezeným a je nedostačující v řadě případů [12].

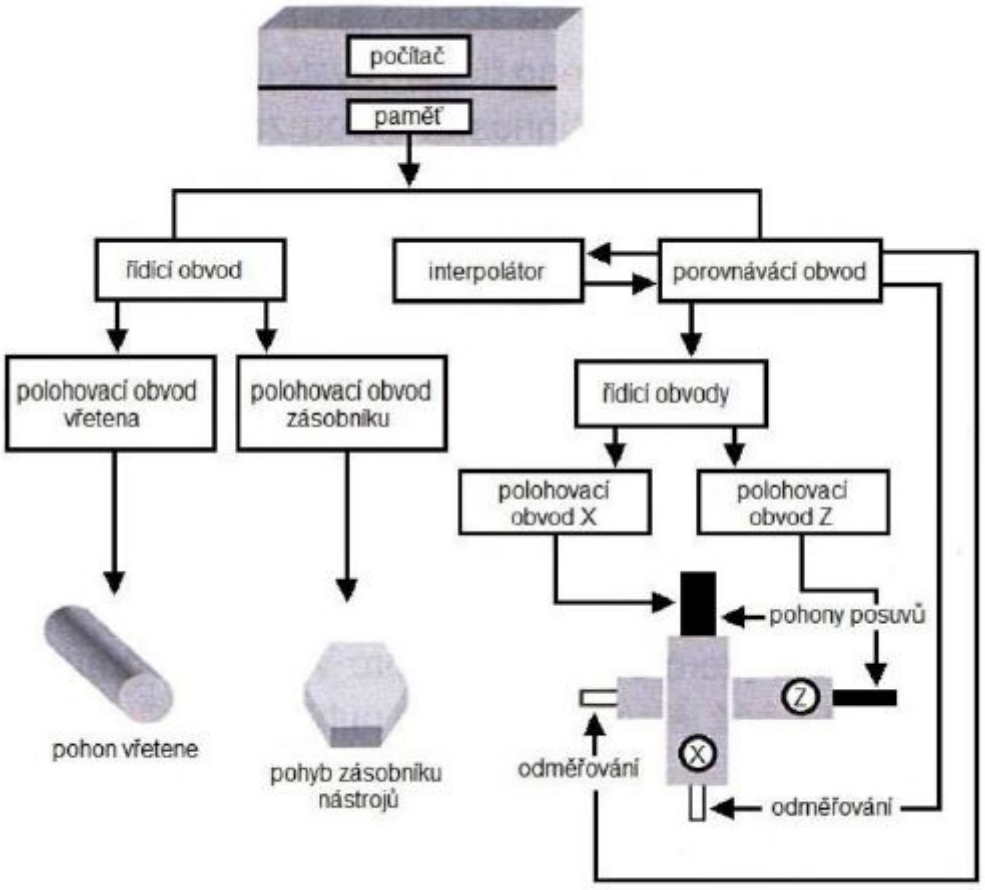
V posledních dvaceti letech parametrické programování prošlo velkým vývojem a dnes už je na velmi pokročilé úrovni. Díky tomu se změnil způsob programování CNC a otevřely se možnosti, které jsou často limitovány pouze znalostmi a kreativitou programátora [12].

1.1.5 Srovnání použitých metod programování

Každý způsob programování má svá pro a proti. Proto se nedá obecně určit nejvhodnější metoda. Parametrické programování však poskytuje nástroje pro zvýšení efektivity, které nejsou součástí ani jednoho z výše zmiňovaných způsobů programování. Umožňuje tvorbu programů, které mohou být využity pro obrábění tvarově příbuzných komponent, dále umožňuje tvorbu soustružnických nebo frézovacích cyklů. Díky logickým operacím se program často umí rozhodovat sám bez zásahu obsluhy stroje.

1.2 Řídicí systémy CNC strojů

Řídicí systém stroje má za úkol dle pokynů zadaných programátorem, převézt každý příkaz programu na mechanickou činnost stroje. Je to v podstatě počítač, který řídí jednotlivé složky stroje, jako jsou např. pohony, posuvy, chlazení a jiné pomocné funkce, atd. Křivku zadanou v programu musí interpolovat, nadělit a poslat k jednotlivým pohonům CNC stroje. Na obr. 1.4 je možno vidět schéma řídicího systému, jeho komponent a propojení [11, 22].



Obr. 1.4 Schéma řídicího systému [22].

Mezi nejvyužívanější ŘS patří Sinumerik, Heidenhain a Fanuc, které jsou zároveň nejvýznamnějšími producenty na evropském trhu a budou popsány v následujících podkapitolách. Další, velmi využívané ŘS jsou např. EMCO, ŘS Mazatrol od společnosti MAZAK.

1.2.1 Sinumerik

Sinumerik je řídicí systém CNC strojů německé společnosti Siemens AG. V současné době jsou nabízeny tři řady ŘS: Sinumerik 808D, 828D a 840D sl. Vzhled operačních panelů jednotlivých řad je na obr. 1.5. **Sinumerik 808D** je verze ŘS, která je určena pro nenáročné aplikace (základní frézování a soustružení). **Sinumerik 828 D** je systém vhodný pro standardní soustružnické a frézovací stroje. Obsahuje nejen možnost programování dle ISO normy, ale také dílenské programování (ShopTurn a ShopMill). Praktická část této

diplomové práce je uskutečněna na systému **Sinumerik 840D sl**, proto mu bude věnována zvýšená pozornost [18].

Obr. 1.5 Vzhled operačních panelů jednotlivých řad systému Sinumerik [20].

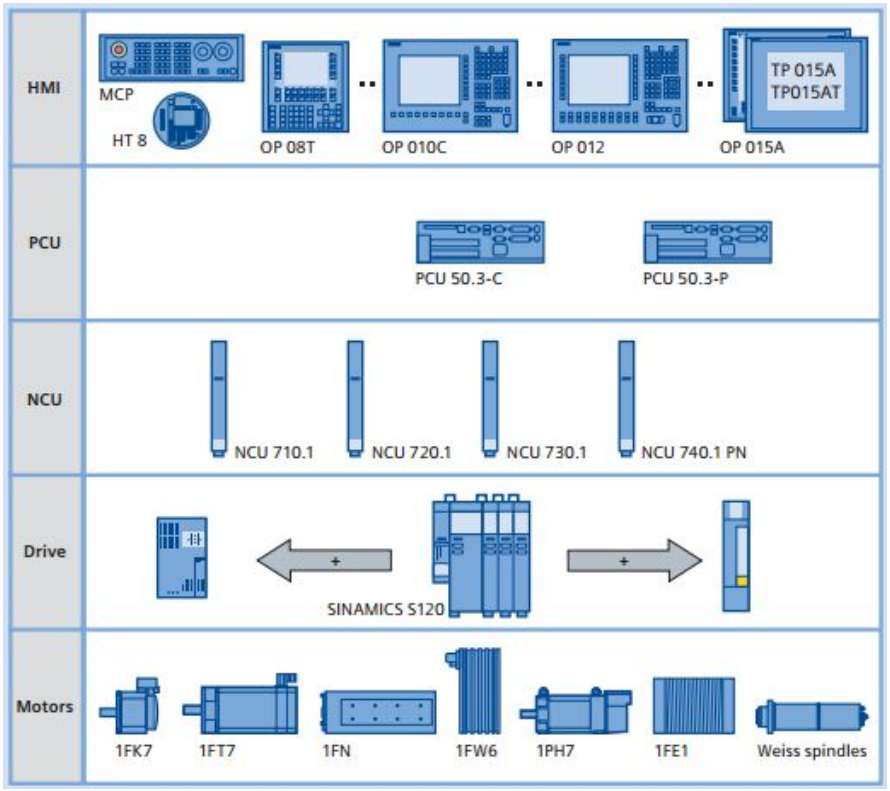
Sinumerik 840D sl

Sinumerik 840D sl je stavebnicový (modulární) systém, reprezentován výrobcem v současnosti jako nejlepší z nabízených řad. Dokáže řídit až 93 os/vřeten v 30 obráběcích kanálech. ŘS prošel v posledních letech vývojem a jeho původní verze PowerLine, která mněla pohonnou soustavu SIMODRIVE, byla nahrazena verzí SolutionLine, která se dodává s pohonnou soustavou SIMATICS. Z toho důvodu přibylo označení „sl“, které původně v názvu této řady nebylo [18, 20, 21].

Tento systém může být sestaven přesně dle požadavků zákazníka a výrobce stroje. Existuje několik možných hardwarových konfigurací, viz obr. 1.6. Na základě požadavků na vlastnosti stroje jsou vybrány jednotlivé komponenty, aby daná konfigurace stroje nejlépe vyhovovala potřebám výrobce i zákazníka [20, 21].

Hlavní komponenty CNC stroje týkající se ŘS Sinumerik 840D sl jsou [20, 22]:

HMI (Human-Machine interface)	rozhraní ovládacích panelů,
PCU (Programmable central unit)	počítač,
NCU (Numerical control unit)	řídicí jednotka,
Drive	výkonová elektronika (frekvenční měniče servomotorů),
Motors	motory jednotlivých pohonů.



Obr. 1.6 Ukázka HW komponent stroje s ŘS Sinumerik 840D sl [21].

1.2.2 Fanuc

Fanuc je japonská společnost. Zabývá se automatizací a řízením průmyslu, zejména řízením průmyslových robotů a CNC obráběcích strojů. Stejně jako ostatní výrobci ŘS i Fanuc nabízí několik řad. Výrobce stroje má na výběr z kompaktní řady (0i), kde v základu už je vše podstatné a konfigurace je předdefinovaná nebo může použít modulární řady (30i, 31i, 32i) a konfiguraci provézt sám. Modulární řady se hodí ke složitějším verzím strojů, ale pro vhodné sestavení ŘS je zapotřebí dobré specifikace požadavků ze strany zákazníka [26]. Na obr. 1.7 je ukázka operačních panelů ŘS Fanuc.

Obr. 1.7 Ukázka operačních panelů ŘS Fanuc [26].

1.2.3 Heidenhain

Heidenhain je německá společnost. Většina ŘS nabízených touto společností se zaměřuje zejména na frézování, ale nejnovější systém **TNC 640** je určen pro frézování i soustružení. Na obr. 1.8 je ukázka operačního panelu ŘS Heidenhain TNC 640. Produkty nabízené společností Heidenhain, jako např. různé snímače, dotyková měřidla, dotykové sondy apod., jsou spjaty s konstrukcí CNC strojů a jsou určeny do provozu, kde jsou kladeny nejvyšší nároky na přesnost [23].

V současnosti jsou nabízeny tyto řady ŘS [23]:

TNC 620, TNC 320, iTNC 530, TNC 128 - frézování případně vrtání a vyvrtávání

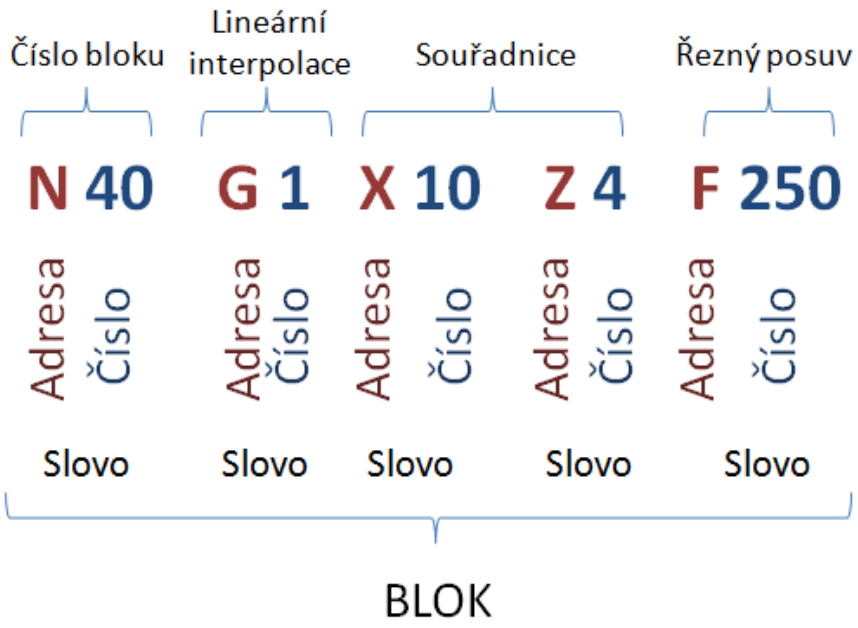
TNC 640 - univerzální systém pro frézování i soustružení

CNC PILOT 640, MANUALplus 620 - soustružení

Obr. 1.8 Ukázka operačního panelu ŘS Heidenhain TNC 640 [23].

1.3 Struktura NC programu

Jak už bylo výše zmíněno, NC program se skládá z bloků, které tvoří jednotlivé řádky. Bloky se skládají ze slov a slova z adresy a čísla jak je možno vidět na obr. 1.9. V zápisu není vyžadováno přesné pořadí adres, ale z důvodu eliminace chyb je doporučováno toto seřazení: N G X Y Z F S T D H M. V jednom bloku nesmí být použity dvě adresy ze stejné skupiny. Některé adresy mohou zůstat neobsazeny buďto z důvodu, že je není třeba zadávat nebo z důvodu modality – hodnota funkce se přenáší z předchozího bloku, a tedy ji není třeba znovu definovat [5, 6].



Obr. 1.9 Ukázka struktury bloku.

Mezi jednotlivými ŘS jsou malé rozdíly v zápisu některých částí programu. Např. ŘS Fanuc začíná program znakem procenta (%) a na druhém řádku následuje jméno (:0001). Sinumerik nic takového nemá. V této kapitole bude popsána struktura programu, která bude korespondovat s ŘS Sinumerik, ale obecně je až na malé odchylky stejná u všech ŘS.

1.3.1 Začátek programu (hlavička programu)

Jako hlavička programu jsou označovány bloky, které jsou uvedeny ještě před pohybovými bloky pro výrobu kontury. Tyto bloky nesou informace o výměně nástroje, korekčních parametrech, pohybu vřetena, regulaci pohybu a o geometrických parametrech, jako je např. posunutí počátku [10].

Typickým příkladem je následující kód:

```
N10 G54 G18 G71 G95 G90
N20 T1 D1
N30 G96 S170 M4 M8
N40 LIMS=2500
```

Blok N10: nastavení základních G-funkcí, podrobněji v kapitole 1.3.5,

Blok N20: T1-volba nástroje, D1-vyvolání korekčních parametrů nástroje,

Blok N30: G96-konstantní řezná rychlost, S170-otáčky, M4-vřeteno se otáčí vlevo, M8-zapnutí chladicí kapaliny,

Blok N40: LIMS=2500 meze otáček na 2500 min^{-1}.

1.3.2 Výměna nástroje

U řetězových, plošných a diskových zásobníků probíhá výměna nástroje za pomoci příkazů T a M. Na soustruzích, u revolverových zásobníků je výměna nástroje provedena pouze příkazem T [10].

T - příkaz k vyhledání nástroje v zásobníku,

M - pokyn k výměně nástroje ve vřetenu.

Příklad:

```
...
N110 T2 M6      ; Výměna nástroje z pozice zásobníku č. 2
N130 G0 G90 G56 X50 Z5 S2000 M3; S2000-nastavení otáček
N140 D2         ; Délková kompenzace nástroje
N150 G0 X20     ; Rychloposuv nástroje do bodu X20 Z5
...
```

1.3.3 Ukončení programu

Před ukončením programu je zapotřebí dostat nástroj do výchozí polohy a případně vypnout chlazení. Najet nástrojem do referenční polohy a vypnout vřeteno. Teprve potom může být program vypnut a stroj zastaven [6].

Běžné ukončení programu může vypadat například takto [10].

```
...
N110 G0 X130 Z5      ; Přejezd do výměny nástroje
N130 M5 M9           ; Vypnutí otáček a chlazení
N140 M30             ; Konec programu
...
```

Poslední blok programu obsahuje příkaz určený pro konec programu: M2, M30 příp. M17-konec podprogramu [10].

1.3.4 Základní funkce programování NC strojů

V tabulce 1.1 je uveden seznam základních funkcí, a jejich význam, používaných u programování NC strojů.

Tab. 1.1 Význam vybraných adres seřazených v doporučeném pořadí [6, 10].

Adresa	Příklad	Funkce
N	N10	Číslo bloku (číslo řádku), pro lepší orientaci
G	G28	Přípravná funkce. Více v kapitole o G-funkcích
X, Y, Z	X10.5	Souřadnice
A, B, C	A90	Rotace kolem základních translačních os A-kolem osy X, B kolem Y, C kolem Z. Jednotky ve stupních nebo radiánech.
R	R100	Poloměr oblouku
R	R0=250	Volitelný parametr
F	F200	Posuv, většinou v jednotkách [mm/min] nebo [mm]
S	S1000	Otáčky vřetene nebo konstantní řezná rychlost, většinou [1/min], [m/min]
T	T12	Číslo nástroje
D	D12	Korekce nástroje
M	M3	Pomocná strojní funkce. Např. M3-směr otáčení vřetena vpravo.

1.3.5 G-funkce

G-funkce jsou přípravné funkce, kterými se definuje např. dráha (G0, G1, G2, G3), volba roviny (G17, G18, G19), nájezd do referenční polohy, absolutní či inkrementální souřadnice, vstup v milimetrech nebo v palcích, korekce nástroje a další. Seznam vybraných G-funkcí je v tab. 1.2.

Tab. 1.2 Seznam vybraných G-funkcí [10, 11].

G-funkce	Význam
G0	Nájezd z bodu do bodu rychloposuvem
G1	Lineární interpolace – pracovní posuv
G2	Kruhová interpolace ve směru hodinových ručiček
G3	Kruhová interpolace proti směru hodinových ručiček
G4	Časová prodleva
G17, G18, G19	Volba roviny XY, ZX, nebo YZ
G20, G21	Vstup v palcích, vstup v milimetrech
G32	Řezání závitu
G41, G42	Korekce zaoblení špičky nástroje (nástroj zprava, zleva)
G40	Zrušení korekce zaoblení špičky nástroje
G54-G57	Nastavení posunutí počátku souřadnic
G90	Programování v absolutních souřadnicích
G91	Programování v inkrementálních souřadnicích
G93	Časově inverzní posuv [1/min]
G94	Lineární posuv [mm/min], [palce/min] nebo [stupně/min]
G95	Otáčkový posuv [mm] nebo [palce]

Většina G-funkcí má **modální** platnost. To znamená, že jejich funkce je platná i pro další řádky a nemusí se psát znovu. Naopak příkazy, které mají pouze **blokovou** platnost, platí pouze na řádku, v němž jsou naprogramovány.

1.4 Příklady parametrického programování v Sinumeriku 840 D sl / 828 D

1.4.1 Proměnné

ŘS Sinumerik dává k dispozici několik druhů proměnných, díky kterým je možno ve spojení s matematickými funkcemi sestavovat cykly a jiné komplexní výrobní programy. Prvním typem jsou **systémové proměnné**, které jsou předem definované v systému a jsou uživateli k dispozici. Příkladem mohou být strojní parametry. Druhým základním typem jsou **uživatelské proměnné**, které jsou dále rozděleny na **předdefinované uživatelské proměnné** a **uživatelem definované proměnné** [2].

Rozdělení proměnných [2]:

1) Systémové proměnné
   - proměnné předběžného zpracování,
   - proměnné hlavního zpracování.
2) Uživatelské proměnné
   - předdefinované uživatelské proměnné
     - R – parametry,
     - linkové proměnné,
   - uživatelem definované proměnné
     - Lokální uživatelské proměnné (LUD),
     - Programově globální uživatelské proměnné (PUD),
     - Globální uživatelské proměnné (GUD).

R-parametry

V systému je k dispozici až 99 R-parametrů. Zápis R-parametru může vypadat takto:

```
R17 = SIN45
```

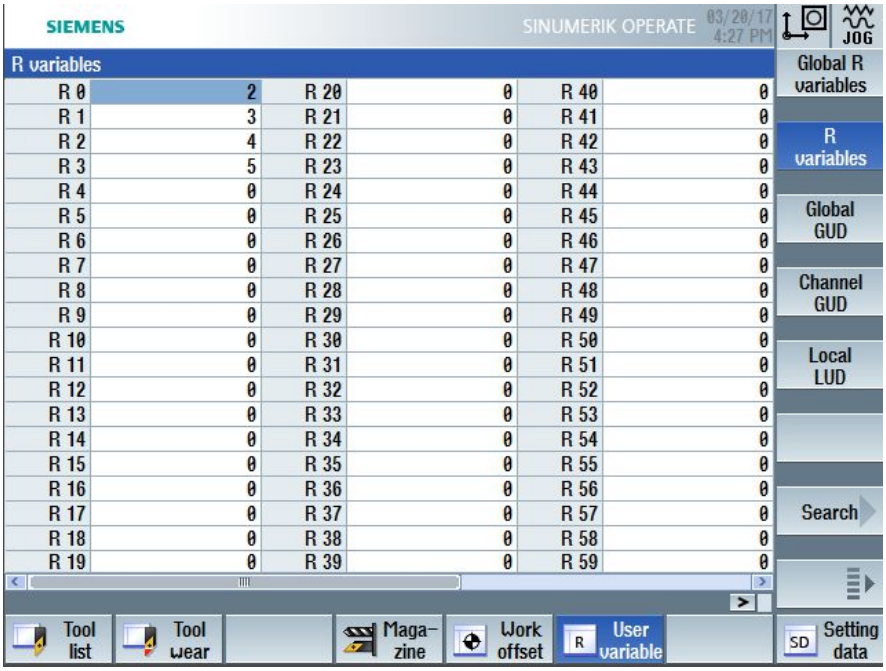
U programování synchronních akcí se přidává znak „$“ ($R17=…)

Na obr. 1.10 je znázorněn zápis R-parametrů v Sinutrainu a možnost zápisu dalších uživatelských parametrů (např. GUD) [2].

Uživatelem definované proměnné

Tyto proměnné si může programátor nadefinovat sám pomocí příkazu `DEF`. Existují tři typy uživatelem definovaných proměnných. Dělí se dle oblasti, ve které se proměnná zobrazuje. LUD proměnné jsou v programu, který není hlavní, ale je vyvolán jiným programem, čímž se tyto proměnné založí. S ukončením daného programu jsou vymazány. PUD proměnné jsou naopak založeny v hlavním programu a jsou vymazány s ukončením tohoto programu. GUD proměnné jsou globální, je k nim přístup ve všech programech a jsou zachovány i po vypnutí systému [2].

U definice uživatelských proměnných se musí také nadefinovat datový typ proměnné. Např. INT – celočíselné hodnoty se znaménkem, REAL – reálné číslo, BOOL – boolovská hodnota kde 1=true, 2=false a další [2].



R 0	2	R 20	0	R 40	0
R 1	3	R 21	0	R 41	0
R 2	4	R 22	0	R 42	0
R 3	5	R 23	0	R 43	0
R 4	0	R 24	0	R 44	0
R 5	0	R 25	0	R 45	0
R 6	0	R 26	0	R 46	0
R 7	0	R 27	0	R 47	0
R 8	0	R 28	0	R 48	0
R 9	0	R 29	0	R 49	0
R 10	0	R 30	0	R 50	0
R 11	0	R 31	0	R 51	0
R 12	0	R 32	0	R 52	0
R 13	0	R 33	0	R 53	0
R 14	0	R 34	0	R 54	0
R 15	0	R 35	0	R 55	0
R 16	0	R 36	0	R 56	0
R 17	0	R 37	0	R 57	0
R 18	0	R 38	0	R 58	0
R 19	0	R 39	0	R 59	0

Obr. 1.10 Zápis R-parametrů v Sinutrainu V4.7 [13].

1.4.2 Podprogramy

Podprogramy jsou velmi důležitou součástí programování vesměs všech programovacích jazyků, včetně programování CNC strojů. Využívají se především, je-li nějaká část programu, která se opakuje. Tento úsek programu se napíše do podprogramu, který je následně vyvoláván z hlavního programu.

V systému Sinumerik může být jakýkoliv výrobní program spuštěn jako hlavní nebo vyvolán jiným programem jako podprogram. Podprogramem se tedy označují jakékoli programy, které jsou vyvolány programem jiným [2].

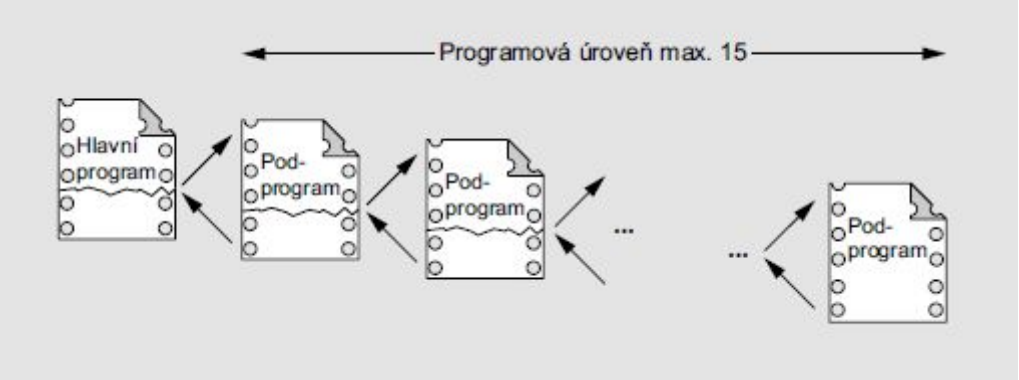
Výhodou psaní určitých částí kódu do podprogramů je lepší přehlednost a čitelnost v programu. Programátor si může vytvořit vlastní knihovny programů, specifické pro jeho výrobní sortiment. Další výhodou je úspora místa v paměti [2].

U psaní názvů smí být použito maximálně 31 znaků. První dva musí být písmena a následující znaky mohou být libovolnou kombinací číslic a podtržení (“_“). Velká a malá písmena v jazyku NC systému Sinumerik nejsou rozlišována. Název programu je dále rozšířen o předponu a příponu. Předpona programu je (_N_). Přípony pro hlavní programy jsou (_MPF) a pro podprogramy (_SPF). U volání podprogramů programátor může využít jakoukoliv kombinaci názvu programu, přípony a předpony [2].

Např. když se podprogram bude jmenovat „PP_HRUB“ tak jeho názvy mohou vypadat následovně:

1) _N_PP_HRUB
2) PP_HRUB
3) PP_HRUB_SPF
4) _N_PP_HRUB_SPF

V každém hlavním programu může být vyvolán podprogram a z něj pak další podprogram. Přičemž každý podprogram pracuje na své vlastní programové úrovni. V současné době je k dispozici 16 programových úrovní. Hlavní program má nejvyšší programovou úroveň 0. První vyvolaný podprogram má úroveň 1 a tak to jde dál až do 15. Hloubku vnoření názorně zobrazuje obr. 1.11 [2].



Obr. 1.11 Hloubka vnoření podprogramů [2].

Když programátor vyvolá podprogram bez udání cesty, tak ŘS hledá v následujících adresářích v daném pořadí [2].

1) Aktuální adresář (adresář volajícího programu)
2) /_N_SPF_DIR/ (adresář globálních podprogramů)
3) /_N_CUS_DIR/ (uživatelské cykly)
4) /_N_CMA_DIR/ (cykly výrobce)
5) /_N_CST_DIR/ (standardní cykly)

1.4.3 Předávání parametrů

Když je podprogram vyvolán, musí mu být přiřazeny parametry, které mohou být dvojího typu. **Formální parametry** jsou definovány při definici podprogramu a následně jsou předány spolu s jejich typem a názvy. Dalším typem jsou **skutečné parametry**, které doplňují do rozhraní programu skutečné hodnoty [2].

Formální parametry – příklad:

```
PROC SUB_1(REAL DELKA, REAL VYSKA); parametry DELKA, VYSKA

N10 G1 X=DELKA       ; pracovní posuv do polohy X=DELKA

N20 Z=VYSKA          ; pracovní posuv do polohy Z=VYSKA

...

N200 RET
```

Skutečné parametry – příklad:

```
N10 DEF REAL DELKA, REAL VYSKA      ; definice parametrů
N20 DELKA=60                        ; přiřazení hodnot parametrům
N30 VYSKA=20
N40 SUB_1(DELKA, VYSKA)  ; vyvolání podprogramu SUB_1 se skutečnými
                         parametry a jejich hodnotami DELKA=60,
                         VYSKA=20
...
N200 M30
```

U volání podprogramu nemusí být vždy předávány všechny parametry, které jsou v podprogramu definovány. Když danému parametru nebude přidána žádná skutečná hodnota, bude mu automaticky přidělena hodnota „0“. Pro jednoznačnost a správnou identifikaci jednotlivých parametrů, musí být dodržena posloupnost a vždy musí být zapsány čárky, které slouží k oddělení jednotlivých parametrů. Pouze v případě, že nebude přidána hodnota posledním parametrům, čárky mohou být vypuštěny [2].

Příklad:

```
/Podprogram/
PROC SUB_1(REAL DIA_1, REAL DIA_2, REAL DIA_3)
...
N200 RET


/Hlavní program/
...
N20 SUB_1(20, 30, 5)
...
M30
```

Další možnosti jak může být zapsán blok N20 pro vyvolání podprogramu:

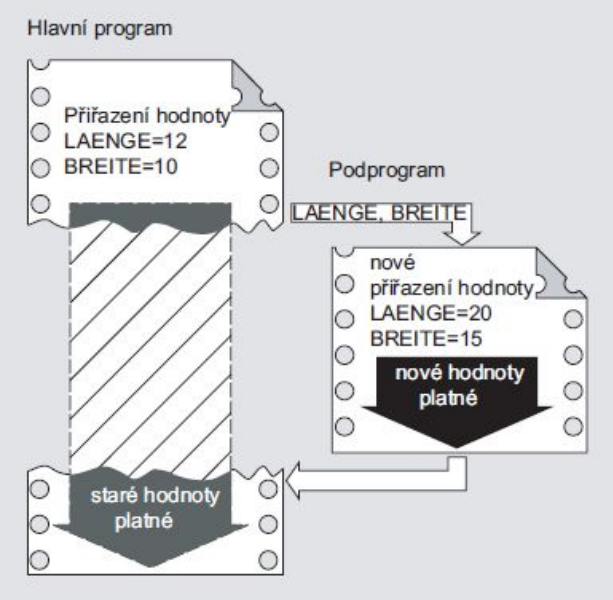
```
N20 SUB_1( , , )
N20 SUB_1(20, ,5)
N20 SUB_1(20,30)
N20 SUB_1( ,30,5)
```

Podprogramu mohou být předány parametry pomocí klíčového slova PROC, za kterým se píše název programu. Následuje výčet všech parametrů, které jsou podprogramem očekávány. Příkaz PROC se musí nacházet na prvním řádku programu. Tento typ předávání parametrů se jmenuje **Call by Value** a nemá žádný zpětný vliv na volající program, viz obr. 1.12 [2].

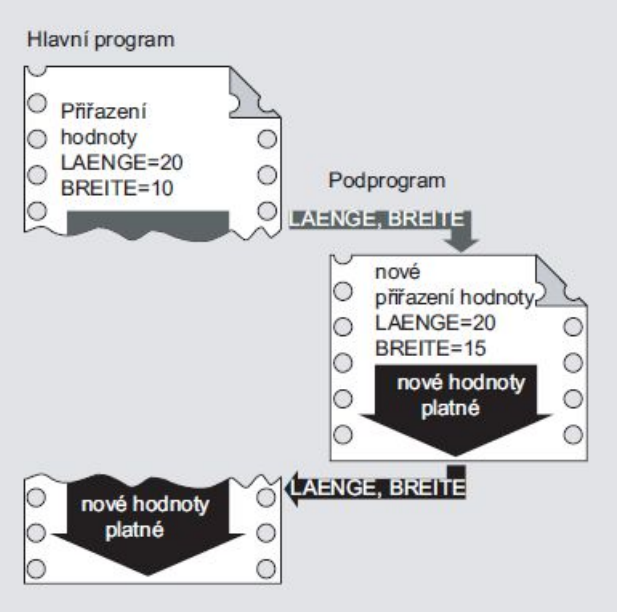
Druhým způsobem je podprogram s předáváním parametrů **Call by Reference**. Tento způsob, na rozdíl od předchozího, má zpětný vliv na volající program, viz obr. 1.13. Zápis se pro definici podprogramu uskutečňuje stejným způsobem až na to, že u výpisu všech podprogramem očekávaných parametrů se připisuje k jednotlivým parametrům klíčové slovo VAR [2].

Příklad:

```
PROC SUB_1(VAR REAL DIA_1, VAR REAL DIA_2, VAR REAL DIA_3)
```



Obr. 1.12 Podprogram s předáváním parametrů Call by Value [2].



Obr. 1.13 Podprogram s předáváním parametrů Call by Reference [2].

1.4.4 Matematické funkce, porovnávací a logické operace

V ŘS je celá řada matematických funkcí od těch základních (+, -, *, /) po složitější (sin(), cos(), MOD – zbytek po dělení) a spousta dalších. U matematických operací jsou priority při zpracování určovány pomocí kulatých závorek a platí obvyklý matematický zápis [2].

Pro formulování podmínek se využívají porovnávací operace. Relační operátory jsou: == rovná se, <> nerovná se, > větší, < menší, >= větší nebo rovno, <=menší nebo rovno. Logickými operátory jsou: AND-logické a, OR-logické nebo, NOT-negace, XOR-logické XOR [2].

Příklad:

```
IF R0>50 GOTOF KONEC
```

nebo využití i logického operátoru

```
IF R0>50 AND R1<100 GOTOF KONEC
```

1.4.5 Řídicí struktury

Standardní zpracování bloků řídicím systémem je dle jejich posloupnosti. Tato posloupnost může být změněna použitím prvků řídicích struktur IF…ELSE, LOOP, FOR, WHILE a REPEAT.

IF, ELSE, ENDIF

Podmínka IF se používá, když je zapotřebí podmínit uskutečnění nějakého bloku. Pokud je podmínka splněna, uskuteční se blok, který následuje po příkazu IF. Jestliže podmínka splněna není, uskuteční se blok, který následuje po příkazu ELSE. Příkaz ELSE není nutné použít v případě, že není žádná alternativa [2].

Příklad:

```
DEF REAL ALFA, REAL BETA, REAL GAMA   ; definice proměnných
ALFA=90                               ; přiřazení hodnot
BETA=45
GAMA=45
N30 IF ALFA+BETA+GAMA<>180            ; jestliže součet úhlů
N40 MSG(„Špatně zadaná geometrie“)    nebude 180, zobrazí se
N45 ELSE                              blok N40 jinak se
N50 MSG(„Geometrie trojúhelníku je    zobrazí blok N50
v pořádku“)
...
N200 M30
```

Programování smyček (LOOP, FOR, WHILE)

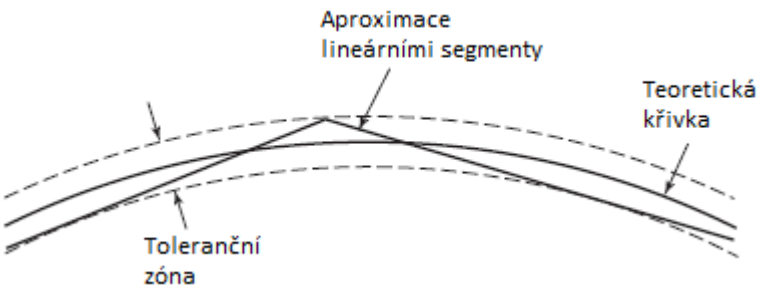
V případě programování nekonečných programů se využívá nekonečná smyčka, která se vyvolá příkazem LOOP a po příkazu ENDLOOP se vrátí na její začátek. Jestliže je počet opakování daného úseku programu předem znám, využívá se smyčka s počítadlem (FOR…TO, ENDFOR). Pro smyčku, která se koná do té doby, dokud je splněna nějaká podmínka, se používá příkaz WHILE, ENDWHILE [2].

Příkladem smyčky s počítadlem může být programování pohybu nástroje po kontuře. Tento pohyb má být opakovaný, ale vždy začínat z jiného bodu, který vzniká pravidelným posunutím souřadnic:

```
...
N20DEF INT VAR_i                          ; definice proměnné typu INT
N30FOR VAR_i=1 TO 50
N40 G0 X=(30+(0,005*VAR_i))Z=2            ; N40-nastavení poč. bodu pohybu nástroje
N50 G1 X=60 Z=(-50-(0,005*VAR_i))         ; N50-nastavení konečného bodu pohybu nástroje
N60 G0 Z2
N70 ENDFOR
...
```

1.4.6 Translační křivky

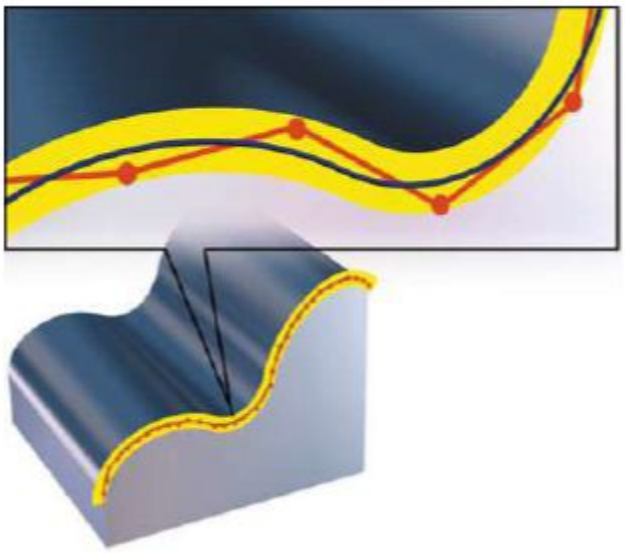
Geometrie reprezentovaná CAD softwarem, není vždy totožná s geometrií, která je produkovaná na CNC stroji. Standardní CNC stroj se umí pohybovat pouze po lineárních nebo kruhových entitách a neumí přímo vytvořit např. elipsu. CAM systémy obvykle exportují danou geometrii také pouze jako lineární aproximace, čímž vznikne dráha nástroje, která je tvořena úsečkami-krátkými segmenty. Tyto segmenty jsou ohraničeny toleranční zónou, viz obr. 1.14. Čím je tolerance menší, tím více segmenty bude křivka proložena, naopak při větší toleranci budou segmenty delší a bude jich méně. Použitím této techniky program naroste, protože každý lineární pohyb vyžaduje vlastní blok se souřadnicemi [11, 24].



Obr. 1.14 Aproximace křivky sérií lineárních segmentů [11].

V přechodech mezi jednotlivými segmenty lineární aproximace dochází ke skokovým změnám rychlosti pohybu nástroje. Skokové změny rychlosti mohou vézt k zhoršení kvality povrchu v důsledku vibrací a zkoseného povrchu. V případě přesného obrábění jsou zde kladeny požadavky na hladké pohyby nástroje [25].

Některé CAM systémy umí konvertovat lineární segmenty do kruhových, čímž se zredukuje počet bloků v programu a vede to k hladšímu povrchu [24]. CNC systémy nabízí možnost vyhlazení pohybu os pomocí interpolačních splajn křivek (např. BSPLINE) nebo pomocí kompresoru bloků. V ŘS Sinumerik se využívá příkazu **COMPCON, COMPCURV** nebo **COMPCAD**, které spustí vyhlazení trajektorie. V návaznosti na specifikovanou toleranci, kompresor vezme sérii NC příkazů, vyhodnotí je a utvoří z nich splajn. Díky tomu se nástroj pohybuje po hladších plochách a vzniká kvalitnější povrch [25]. Na obr. 1.15 je možno vidět vyhlazení nesouvislých bloků pomocí kompresoru.



Obr. 1.15 Vyhlazení nesouvislých bloků proložením splajn křivky za použití kompresoru [25].

1.5 Reprezentace geometrie v PC

V dnešní době je použití výpočetní techniky v oblasti 3D geometrie takřka nevyhnutelné. Díky 3D zobrazení tělesa získáváme spoustu výhod: možnost kontroly, rychlé provedení změn, optimalizace produktu aj. 3D model je možné získat buď samotným návrhem, nebo je možné získat data k vytvoření modelu z již existujícího výrobku pomocí digitalizačních zařízení jako je např. 3D scanner ATOS, který se používá u nás na ústavu [14].

Existuje celá řada modelářů a mnoho různých kritérií, dle kterých mohou být rozděleny. Základní rozdělení programů využívaných ve strojírenství by mohlo být na plošné (Rhinoceros) a objemové modelování (Inventor, Solidworks). Za vyšší třídu programů, které zvládají obojí typ modelování na vysoké úrovni, jsou považovány např. CATIA, NX, CREO [14].

Základní geometrickou entitou jsou křivky, z kterých se skládají plochy, a zároveň se po nich pohybuje nástroj. Objem tělesa se dá vytvořit různými způsoby, které budou nastíněny v této podkapitole [15].

1.5.1 Rozdělení křivek

Křivka je v podstatě soustava parametrů nějaké rovnice, která je zobrazována. Existují tři základní vyjádření křivek [14]:

1) **Explicitní**

   `y=f(x)`

2) **Implicitní**

   `f(x,y)=0`

3) **Parametrické**

   `x=x(t),`

   `y=y(t),`

   `z=z(t),`

   `kde t∈I.`

Běžný způsob zadávání křivek v matematice je za pomoci **explicitních** a **implicitních** rovnic. Tento způsob však není vhodný pro počítačové zobrazování geometrie, ani pro řídící jednotky CNC strojů. Problémy nastanou už v případě směrnice přímky blížící se k nekonečnu, kde se vyskytují numerické chyby. Dalšími problémy jsou [15]:

- neomezená geometrie (místo úsečky, přímka),

- když je změna souřadného systému, bude se lišit i rovnice,

- pro jednu proměnnou mohou křivky nabývat vícero hodnot,

- u průniku dvou jednoduchých těles (např. válců) mohou vznikat zbytečně komplexní rovnice. V takovém případě by bylo snazší křivku interpolovat za pomoci série bodů.

1.5.2 Parametrické vyjádření křivek

Základním typem křivek jsou křivky polynomiální. Tyto polynomiální křivky se dále seskupují a tvoří křivky po částech polynomiální. V praxi se nejvíce používají křivky třetího stupně, tzv. kubiky. Za použití kubik lze obsáhnout velké spektrum tvarů. Jsou nenáročné na výpočet, snadno se s nimi manipuluje a je možné u nich zajistit spojitost C2, která bývá často u modelování v CAD-systémech požadována. V případě křivek vyššího stupně mohou vznikat oscilace křivek, které jsou nežádoucí a způsobují velkou náročnost výpočtu [14].

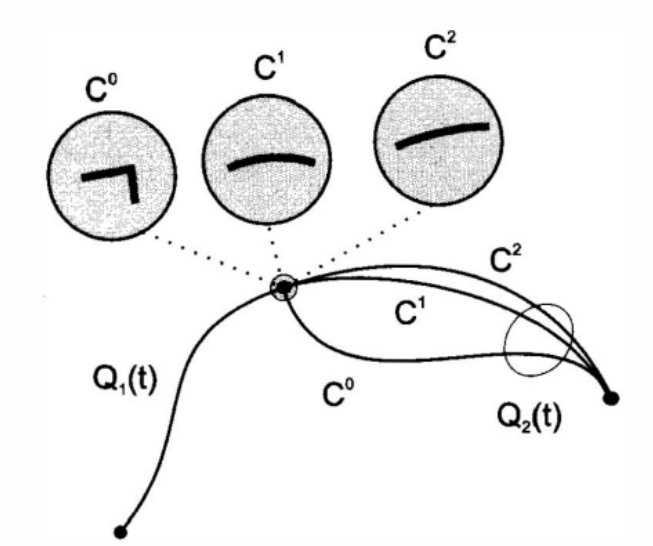
Křivky po částech polynomiální jsou navazovány oblouky. Důležitým faktorem je zde **spojitost**. Křivka je spojitá, jestliže je spojitá ve všech svých bodech, včetně navazujících bodů. Pokud je křivka spojitá a ve všech svých bodech je spojitá i první derivace křivky, nazývá se **hladká křivka**. Existují dva typy spojitostí [14, 17]:

1) **Parametrická spojitost (C0, C1, C2,…)**

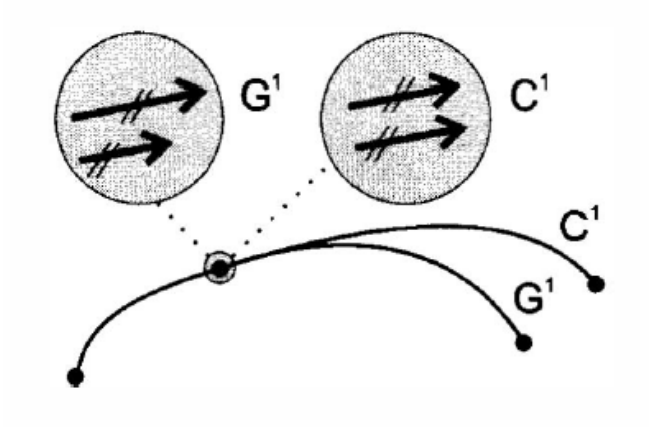
   Na obr. 1.16 jsou znázorněny jednotlivé třídy spojitosti. Pro spojitost třídy C0 stačí pouze, aby byl koncový bod prvního segmentu $Q_1(t)$ počátečním bodem druhého segmentu $Q_2(t)$. Spojitost C1 je zajištěna pokud se rovnají tečné vektory v koncových bodech obou segmentů. Pro C2 spojitost je požadována rovnost vektorů první a druhé derivace [17].

2) Geometrická spojitost

Spojitost C1 implikuje G1, obráceně to však neplatí. Rozdíl mezi parametrickou a geometrickou spojitostí je na pohled těžko rozpoznatelný, je však patrný v případě, že se jedná o křivku, po které se pohybuje nějaký objekt. Např. rozdíl mezi spojitostí C1 a G1 je takový, že C1 zajišťuje v bodě, kde křivky na sebe navazují, stejnou polohu a rychlost objektu. V případě spojitosti G1 bude zaručena poloha, rychlost bude mít stejný směr, ale změna hodnoty rychlosti se bude měnit skokem. Z toho tedy vyplývá, že vhodnější použití v počítačové grafice je C-spojitost. Zobrazení geometrické a parametrické spojitosti popisuje obr. 1.17, kde je možno vidět, že směry vektorů jsou v obou případech stejné, ale u G1 spojitosti je velikost vektoru rychlosti jiná [17].



Obr. 1.16 Názorná ukázka C0, C1 a C2 spojitostí [17].



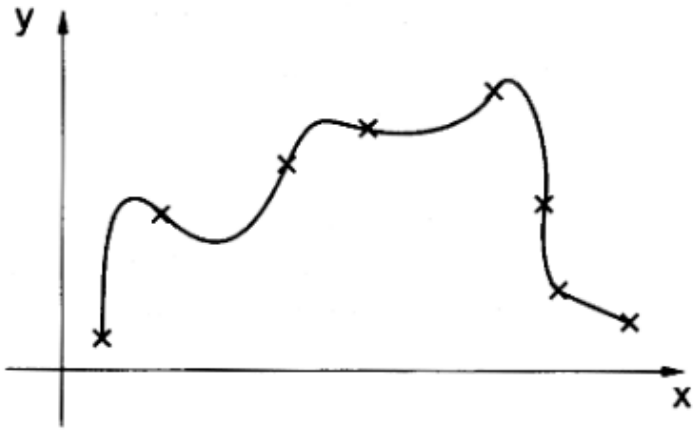
Obr. 1.17 Rozdíl mezi parametrickou a geometrickou spojitostí [17].

1.5.3 Modelování křivek

Křivka je obvykle definovaná několika řídicími body, které tvoří řídicí polygon. Z polohy bodů řídícího polygonu se určuje tvar a průběh křivky. Interpretace řídících bodů může probíhat dvěma způsoby [14]:

1) **Interpolace**

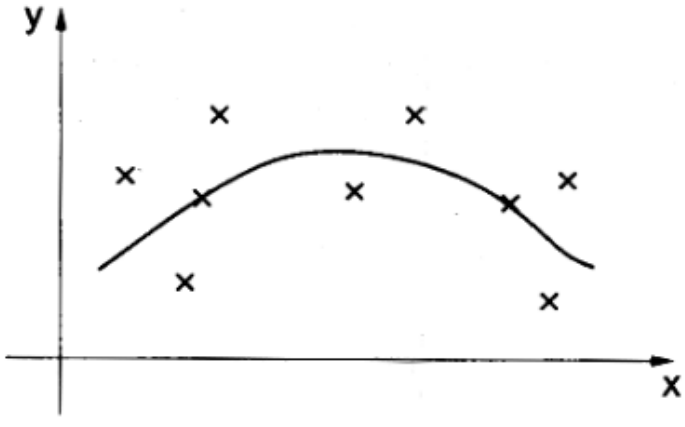
   Křivka probíhá danými body, viz obr. 1.18. Mezi nejznámější interpolační křivky patří: Lagrangeova interpolační křivka, Hermitovské kubiky a Catmull-Rom splajny [14, 17].



Obr. 1.18 Interpolace-křivka probíhá danými body [17].

2) **Aproximace**

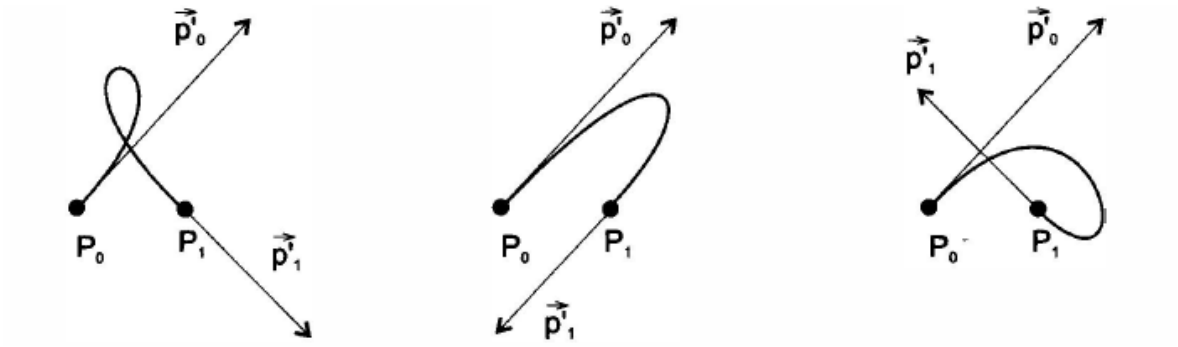
   Křivka nemusí probíhat řídicími body, ale je jimi určena viz obr. 1.19. Mezi nejpoužívanější aproximační křivky patří: Beziérovy, Coonsovy, B-splajn křivky, NURBS.



Obr. 1.19 Aproximace-křivka je určena řídicími body, ale nemusí jimi procházet [17].

Hermitovské kubiky

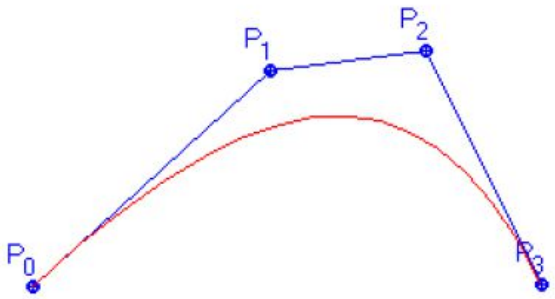
Někdy jsou také označovány jako Fergusonovy kubiky. Křivka je tvořena dvěma řídicími body a tečnými vektory, které vycházejí z daných bodů, viz obr. 1.20. Poloha křivky je řízena body. Míra vyklenutí křivky je dána velikostí a směrem vektorů [15, 17].



Obr. 1.20 Změna tvaru Hermitovské kubiky v závislosti na velikosti a směru vektorů [17].

Bézierovy kubiky

Tato křivka je tvořena pouze řídicím polygonem bodů, který určuje výsledný tvar křivky. Řídicí polygon se skládá ze čtyř bodů P0, P1, P2 a P3. Bod P0 je počáteční a P3 koncový. Body P1 a P2 tvoří vyklenutí křivky, viz obr. 1.21 [14].



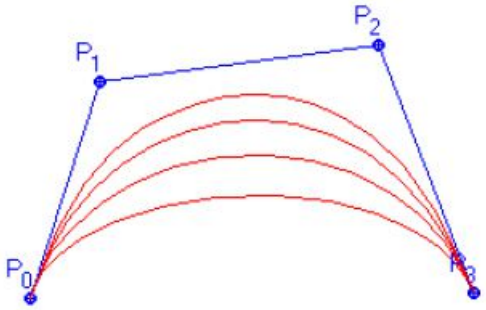
Obr. 1.21 Bézierova kubika [14].

Obecné Bézierovy křivky

Zobecněním Beziérovy kubiky vznikne obecná Beziérova křivka. Není tvořena pouze čtyřmi body jako Beziérova kubika, ale libovolným počtem bodů. Křivka n-tého řádu vzniká z n+1 bodů. Výhodou křivky je snadné hladké napojení [14, 15].

Racionální Bézierovy křivky

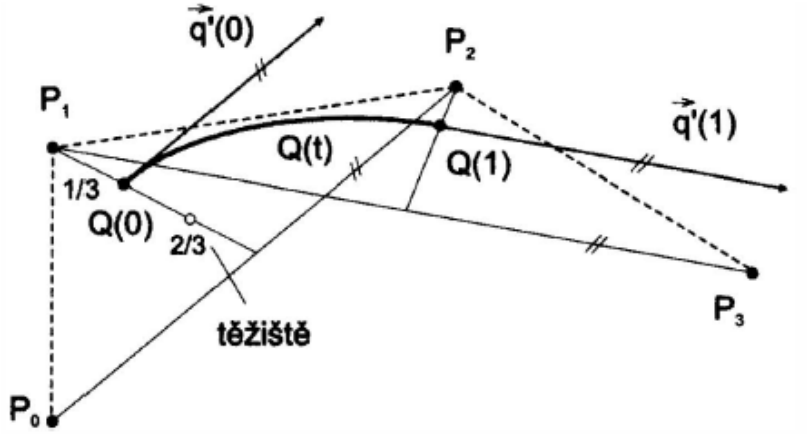
Jestliže se k řídicím bodům Bézierovy kubiky přiřadí nezáporné reálné číslo, které udává váhu bodu, taková křivka se nazývá Racionální Bézierova křivka. Výhodou takové křivky je, že v případech, kdy je to potřeba, se dá měnit její tvar, aniž by se hýbalo s řídicími body, viz obr. 1.22 [14].



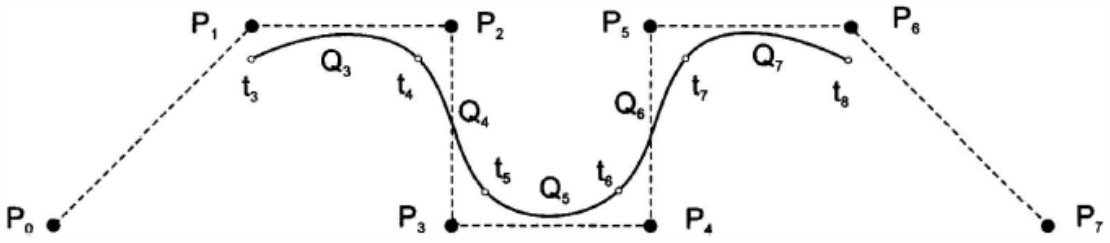
Obr. 1.22 Racionální Bézierovy křivky [14].

Coonsovy kubiky a B-splajny

Coonsova kubika (obr. 1.23) je také, stejně jako Bézierova kubika, zadávána čtyřmi body P0, P1, P2 a P3. Rozdíl je v tom, že krajní body vycházejí z antitěžiště trojúhelníků P0P1P2 a P1P2P3. Výhoda Coonsových kubik spočívá v tom, že když budeme skládat jednotlivé oblouky tak, aby jejich vrcholy byly P0P1P2, P1P2P3 atd., vznikne **uniformní kubický B-splajn** (obr. 1.24). Uniformní kubický B-splajn se někdy také nazývá Coonsův kubický B-splajn. U B-splajnu je ve všech vnitřních bodech zajištěna spojitost druhého řádu. Další výhoda je, že změnou polohy jednoho bodu, se změní pouze čtyři oblouky, na jejichž konstrukci se bod podílí [14, 15, 17].



Obr. 1.23 Coonsova kubika [17].

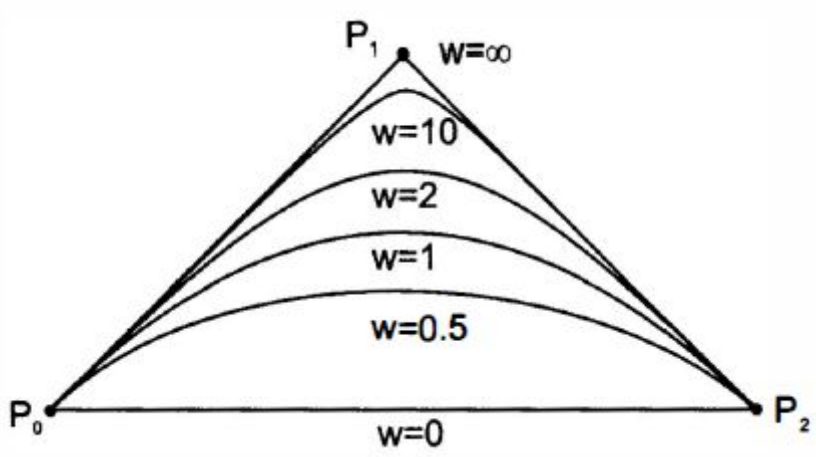


Obr. 1.24 Coonsův kubický B-splajn [17].

NURBS křivky

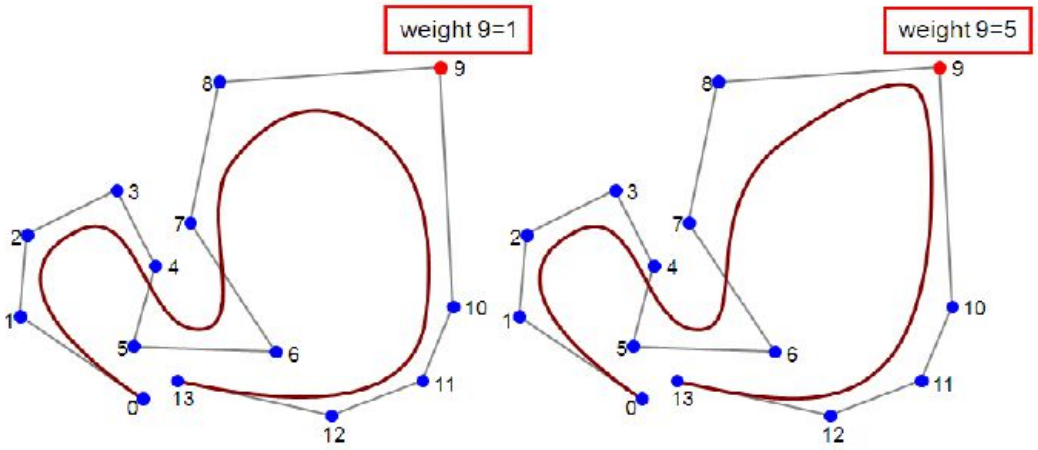
Neuniformní racionální B-splajn křivky *(NURBS - non uniform rational B-spline)* jsou zobecněním B-splajn a Bézierových křivek. Vzdálenost uzlů nemusí být konstantní, což vyjadřuje neuniformnost. Racionalita znamená, že řídicím bodům je přiřazena vlastní váha. Změnou váhy bodu nebo jeho polohy se mění geometrie křivky, ale pouze v okolí daného bodu, viz obr. 1.26 [15, 17].

Na obr. 1.25 je vidět jak se mění křivka změnou váhy bodu P1. Váha bodu P1 (w = 0), tvar křivky neovlivňuje a křivka je v podstatě úsečkou. Pro hodnotu w = 1 je NURBS Bézierovou křivkou a odpovídá neracionální křivce. Čím větší je váha bodu P1, tím více se k němu křivka přimyká. V mezním případě kdy váha bodu bude nabývat nekonečna (w = ∞), křivka začne procházet samotným bodem a ztratí spojitost [14, 17].



Obr. 1.25 Vliv váhy bodu na tvar křivky [17].

NURBS je definována *n+1* body, které tvoří řídicí polygon, uzlovým vektorem *U* a stupněm křivky *p*. Křivka prochází prvním a posledním bodem řídicího polygonu, viz obr. 1.26, kde jsou vyobrazeny obecné NURBS křivky s třinácti řídicími body. Váha bodu P9 se liší, ale změnila se pouze lokální část křivky [14, 15].

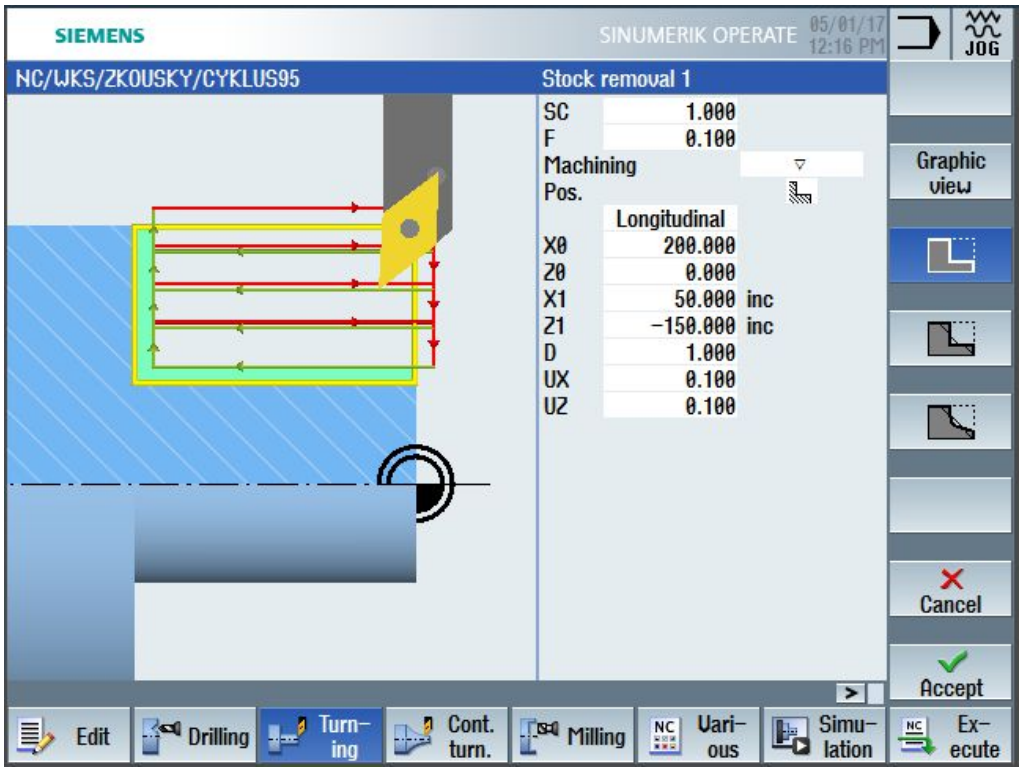


Obr. 1.26 NURBS křivky, jejich průběh a lokální vliv změny váhy bodu [16].

1.6 Grafická podpora v Sinumeriku

K ulehčení práce programátora jsou v řídicím systému stroje standardní technologické cykly pro vrtání, frézování a soustružení. Tyto cykly se dají využít na většinu technologických operací, ale v případě potřeby speciálního cyklu, jsou uživateli k dispozici nástroje, které umožňují tvorbu vhodného cyklu a to včetně grafické podpory [7].

Grafická podpora slouží k tomu, aby obsluha stroje věděla, jaké hodnoty má zadat k příslušným parametrům. Díky grafické podpoře vidí obsluha vše názorně na obrázku a ihned vpisuje hodnoty do tabulky. Tyto hodnoty se poté převezmou do programu, který s nimi počítá. Příkladem je obr. 1.27, kde je grafická podpora cyklu pro soustružení (CYCLE95).



Obr. 1.27 Grafická podpora cyklu pro soustružení (CYCLE95) [13].

1.6.1 Tvorba vlastní grafické podpory

V případě, že máme vlastní cyklus ať už pro obrábění nebo měření součásti, je velmi užitečné mít vytvořenou vlastní grafickou podporu pro daný cyklus, díky které obsluha snadno začne cyklus využívat.

K vytvoření vlastní grafické podpory cyklu jsou zapotřebí čtyři základní kroky:

1) definice proměnných,
2) tvorba softkey uživatelského cyklu,
3) tvorba masky uživatelského cyklu,
4) přiřazení funkcí.

Definice proměnných

Jestliže jsou využity R-parametry, není nutné definovat proměnné. Pokud jsou použity vlastní parametry, musí se definovat v GUD (Global User Data), viz kapitola 1.4.1 o proměnných. Zpravidla se využívá souboru UGUD.DEF, do kterého se definice proměnných zapisuje [7].

Tvorba softkey uživatelského cyklu

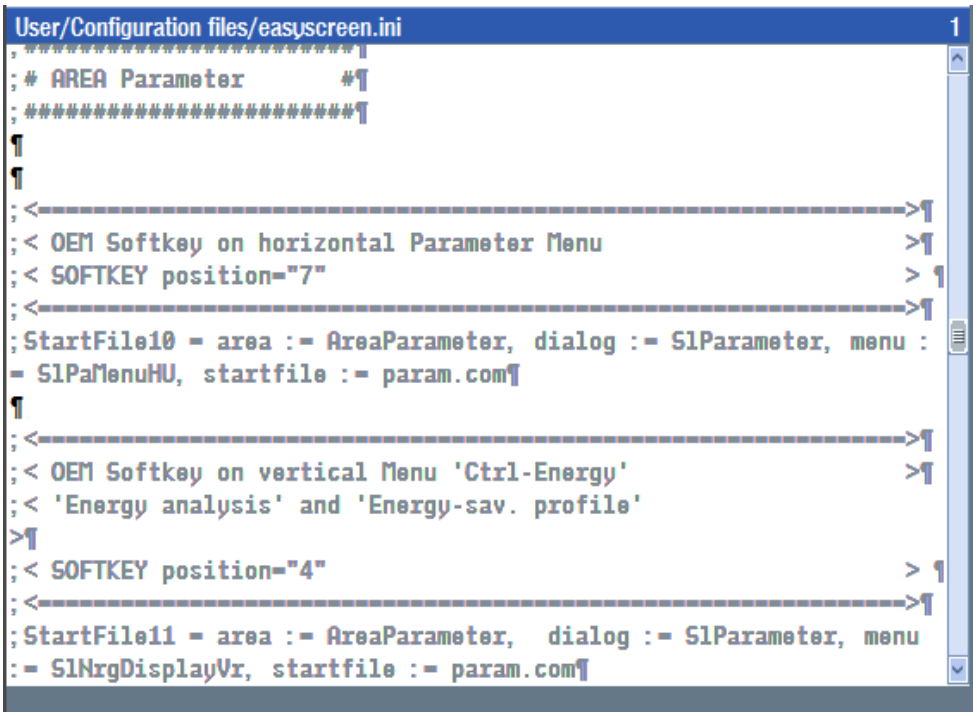
K tvorbě softkey se využívá souboru **easyscreen.ini**, který se nachází v systémových datech (/Data HMI/ Nastavení/ Systém/), kde ho není možné editovat. Proto se soubor zkopíruje do složky- /Data HMI/ Nastavení/ Uživatel/, kde je podle návodu možné soubor editovat. V souboru je seznam softkey, které jsou volné pro uživatele. Každá softkey má svůj kód, který je dále použit [7].

Soubor easyscreen.ini udává cestu, kde se bude softkey nacházet a zároveň odkazuje na soubor, kde je definovaná maska cyklu, viz následující příklad [13]:

```
StartFile15=area:=AreaProgramEdit,     dialog:=SlProgramEdit,
menu:=SlStepStdTurnMenuHU, startfile:=muj_cyklus.com
```

Tento řádek spustí soubor muj_cyklus.com, ve kterém je naprogramovaná maska cyklu a další funkce. Uživatel v celém řádku mění pouze poslední slovo, které je v příkladu zvýrazněno tučným písmem.

V souboru easyscreen.ini je mnoho takových příkazů, které odkazují na určitá místa v prostředí programu Sinutrain, viz obr. 1.28. Tyto příkazy jsou vždy popsány a je před nimi středník. Když jej chce uživatel použít, stačí příkaz zkopírovat a odstranit středník.



Obr. 1.28 Ukázka soboru easyscreen.ini [13].

Tvorba masky uživatelského cyklu a přiřazení funkcí

V soboru s koncovkou „.com“ (v našem případě z předchozího příkladu muj_cyklus.com) se definuje zvolená softkey (název, případně obrázek). Dále se zde definuje maska cyklu (obrázek, text, parametry k vyplnění apod.) Nakonec je třeba vygenerovat z tohoto souboru řádek do programu, který spustí cyklus. To se provede za pomoci příkazu OUTPUT. Cesta souboru s příponou „.com“ je - Systém CF-Card/user/sinumerik/hmi/proj.

Praktický příklad vytvořené vlastní grafické podpory k cyklu se nachází v praktické části diplomové práce v kapitole 3.

2 OBRÁBĚNÍ KOROZIVZDORNÝCH OCELÍ

Korozivzdorné oceli mají zvýšenou odolnost vůči korozi. Základním kritériem korozivzdorných ocelí je minimální obsah chromu 10,5 hm. %. Chrom se v železe rozpouští a tvoří tak ochranný film oxidu chromu [27].

Korozivzdorné oceli mohou být rozděleny do pěti hlavních skupin, dle převažujících prvků v mikrostruktuře viz tab. 2.1.

Tab. 2.1 Rozdělení korozivzdorných a žáruvzdorných ocelí [27].

Ocel	Vlastnosti
Martenzitické oceli	vysoká tvrdost, abrazivní, magnetické, lze tepelně zpracovat
Feritické oceli	měkké čisté železo, magnetické, nelze tepelně zpracovat
Austenitické oceli	nejběžnější struktura, nemagnetické, velké deformační zpevnění
Duplexní oceli	houževnaté, pevné, hůře obrobitelné
PH oceli	PH-precipitačně vytvrzované, tvrdé, pevné, tepelně zpracovatelné

2.1 Vlastnosti korozivzdorných ocelí ovlivňující řezný proces

Velká adheze – z důvodu zvýšené adheze dochází k nárůstku na břitu. Důležité je správné nastavení řezné rychlosti [28].

Nepravidelný tvar utvářených třísek – složitá kontrola dlouhých třísek [28].

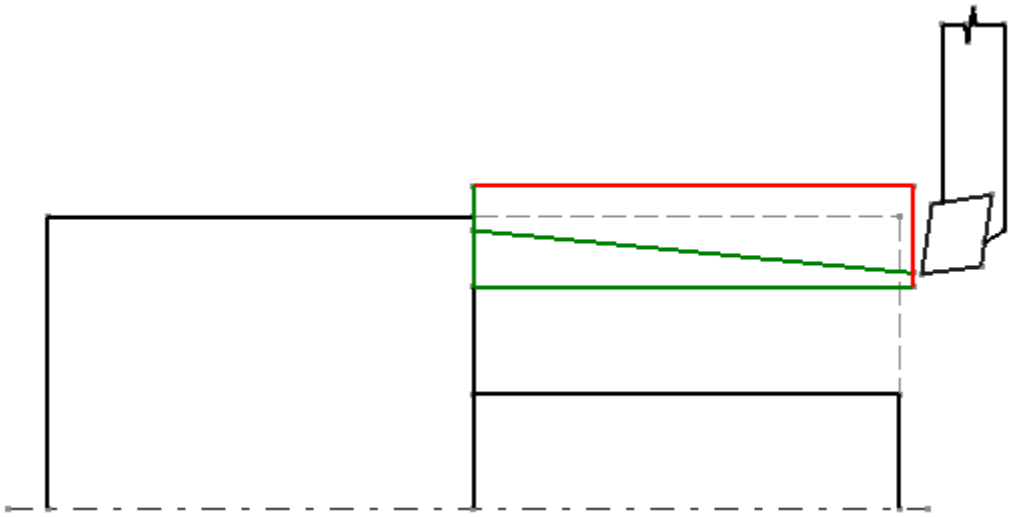
Deformační zpevnění – vysoká náchylnost k deformačnímu zpevnění zapříčiňuje tvorbu tvrdé vrstvy na povrchu obrobku. Rychlé opotřebení nástroje [28].

Nízká tepelná vodivost – teplo neodchází třískami pryč, koncentruje se v oblasti řezu, může vznikat zakalená vrstva [28].

2.2 Soustružení korozivzdorných ocelí

U soustružení korozivzdorných ocelí je důležité zvolit silnou břitovou destičku (odvod tepla) s ostrým, zesíleným břitem (zamezení tvorby nárůstku). Používat houževnaté břitové destičky u hrubování a tvrdé destičky u dokončování. Řez by se měl vézt pod tvrdou povrchovou vrstvou. Mělo by se chladit do správného místa, aby se maximálně omezil vznik tepla [28].

K výraznému zefektivnění soustružení korozivzdorných ocelí může vézt **kolísavá hloubka řezu**. Z důvodu deformačního zpevnění se na povrchu obrobku tvoří tvrdá vrstva. Pokud je hloubka řezu pořád stejná, nástroj naráží na zpevněnou vrstvu pokaždé ve stejném místě a tak se rychle opotřebuje. Výhodou kolísavé hloubky řezu je, že se destička opotřebovává po celé délce břitu rovnoměrně [29], viz obr. 2.1, kde je znázorněna dráha nástroje tak, aby byla zajištěna proměnná hloubka řezu. Zelená barva je dráha pracovního posuvu a červená je dráha rychloposuvu.



Obr. 2.1 Dráha nástroje při proměnné hloubce řezu.

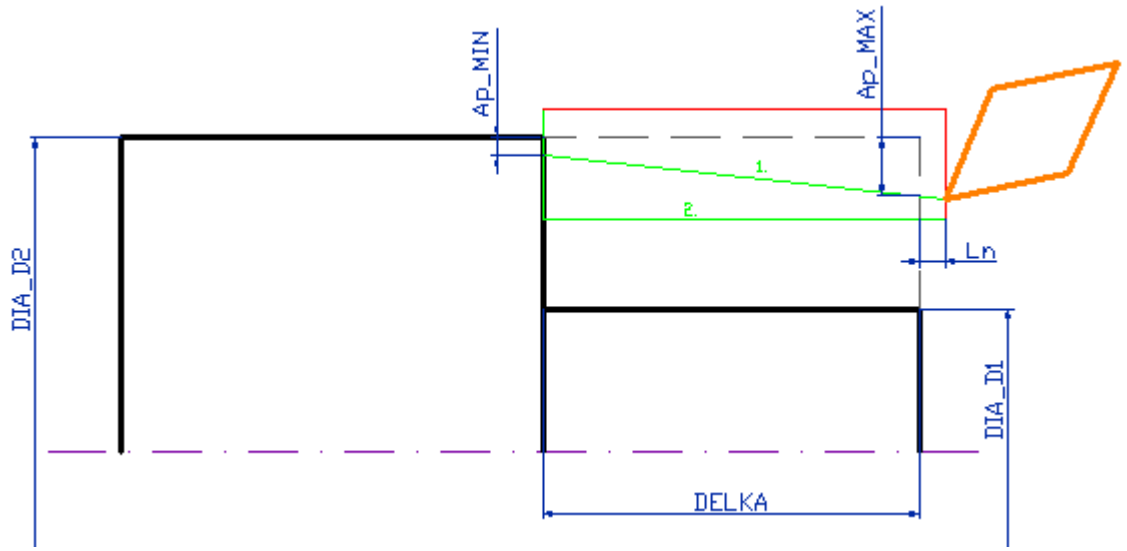
3 NAVRŽENÉ VARIANTY ŘEŠENÍ

Pro praktickou část diplomové práce byly vytvořeny dva parametrické programy, které jsou vyvolávány jako cykly. Tzn., že mohou být vyvolány z jakéhokoli programu. Konkrétně jde o cyklus pro hrubování těžkoobrobitelných ocelí (např. korozivzdorné oceli), který byl nazván „CYCLE2000“ a dalším programem je cyklus pro obrábění šestihranu s názvem „SESTIHRAN“.

3.1 Cyklus pro hrubování těžkoobrobitelných ocelí (CYCLE2000)

Tento parametrický program vznikl na základě poznatků o soustružení korozivzdorných ocelí, viz 2. kapitola. Při klasickém soustružení o stejné hloubce záběru, dochází k opotřebení nástroje pořád na stejném místě a výsledkem je nízká životnost nástroje. Dráha nástroje může být za pomoci CNC naprogramována tak, aby bylo opotřebení nástroje rovnoměrné po celé délce jeho utvařeče (znázornění dráhy je na obr. 2.1).

Cílem bylo, aby byl program schopen hrubovat s proměnnou hloubkou třísky z libovolného průměru DIA_D2 na DIA_D1 v libovolné délce. Všechny geometrické parametry, které jsou uživatelem voleny, jsou na obr. 3.1.



Obr. 3.1 Geometrické parametry, které jsou voleny uživatelem.

Parametry zadávané uživatelem jsou v tabulce 3.1.

Tab. 3.1 Výčet parametrů zadávaných uživatelem cyklu.

Symbol	Jednotka	Popis
DIA_D2	[mm]	průměr polotovaru
DIA_D1	[mm]	požadovaný průměr
DELKA	[mm]	délka obráběné plochy
L_n	[mm]	délka náběhu nástroje
Ap_MAX	[mm]	maximální hloubka záběru
Ap_MIN	[mm]	minimální hloubka záběru
FALZ	[mm]	přídavek v ose Z
FALX	[mm]	přídavek v ose X

FF1	[mm]	posuvová rychlost
DT	[s]	časová prodleva za účelem odlomení třísky

U tvorby cyklu je nutné zajistit, aby při jakýchkoli hodnotách zadaných uživatelem, proběhla operace technologicky správně nebo aby program upozornil na to, že je něco špatně. V tomto případě bylo nejkomplikovanější vyřešit, aby hloubka záběru Ap nebyla nikdy menší než Ap_MIN a nikdy větší než Ap_MAX.

Základní struktura programu cyklu by se dala zjednodušeně napsat takto:

1. Definice parametrů.
2. Pomocné výpočty, podmínky uskutečnění některých operací.
3. Programování drah, po kterých se má nástroj pohybovat.

Třetí část – programování drah byla rozdělena do dvou bloků (viz. kapitola 4 – ukázka programu). V prvním bloku (BLOK_1) jsou naprogramovány dráhy s proměnnou hloubkou záběru. V druhém bloku (BLOK_2) je naprogramováno to, co proměnnou hloubkou záběru už obrobit nešlo – dráha nástroje je vodorovná, nástroj se pohybuje pouze v ose Z.

Následující obrázky (obr. 3.2, 3.3, 3.4 a 3.5) ukazují, jaké varianty mohou nastat, při libovolném zadání Ap_MIN, Ap_MAX, FALX, DIA_D1 a DIA_D2. Pro lepší pochopení a následné zpracování programu byly zavedeny parametry DELTA a VAR_1.

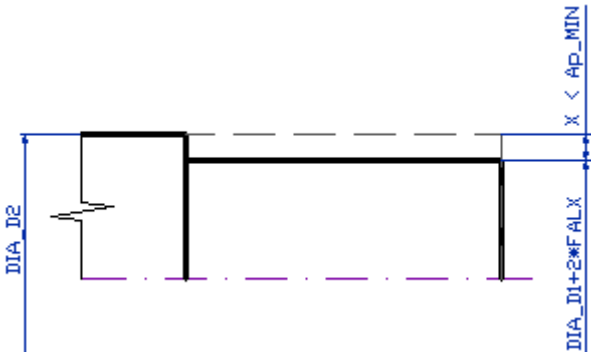
$$DELTA = \frac{DIA_{D2} - (DIA_{D1} + 2 \cdot FALX)}{2} \; MOD \; (Ap_{MIN} + Ap_{MAX}) \tag{3.1}$$

Kde MOD udává zbytek po dělení. Delta tedy udává, kolik zbývá na obrobení po ukončení smyčky, která obrábí s proměnnou hloubkou třísky.

$$VAR_1 = \frac{\frac{DIA_{D2} - (DIA_{D1} + 2 \cdot FALX)}{2}}{Ap_{MIN} + Ap\_MAX} - 1 \tag{3.2}$$

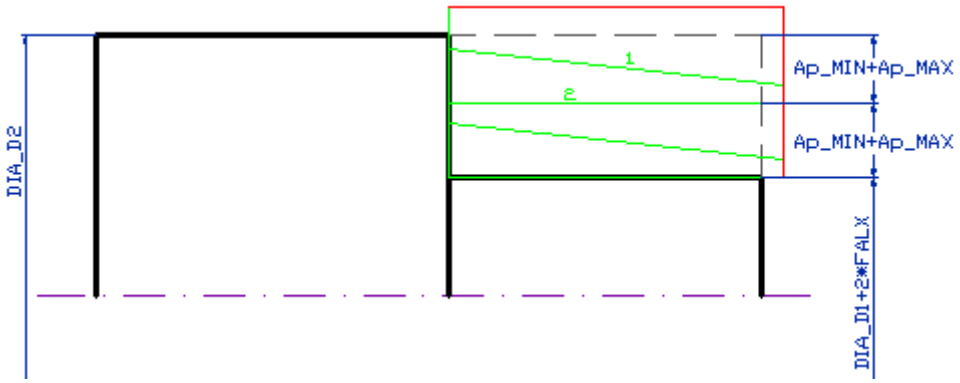
Kde jednička je odečtena proto, že později se s parametrem VAR_1 pracuje. Zaokrouhluje se a v programu nelze zaokrouhlit dolů, pouze nahoru funkcí ROUNDUP. Parametr VAR_1 slouží k tomu, aby se vypočítalo, kolikrát může proběhnout smyčka bloku 1.

První varianta - obsluha zadá příliš malý rozdíl mezi DIA_D2 a DIA_D1 s přídavkem FALX tak, že hloubka záběru by byla menší než Ap_MIN. V takovém případě se program ukončí a vypíše chybovou hlášku. Případ je znázorněn na obr. 3.2.



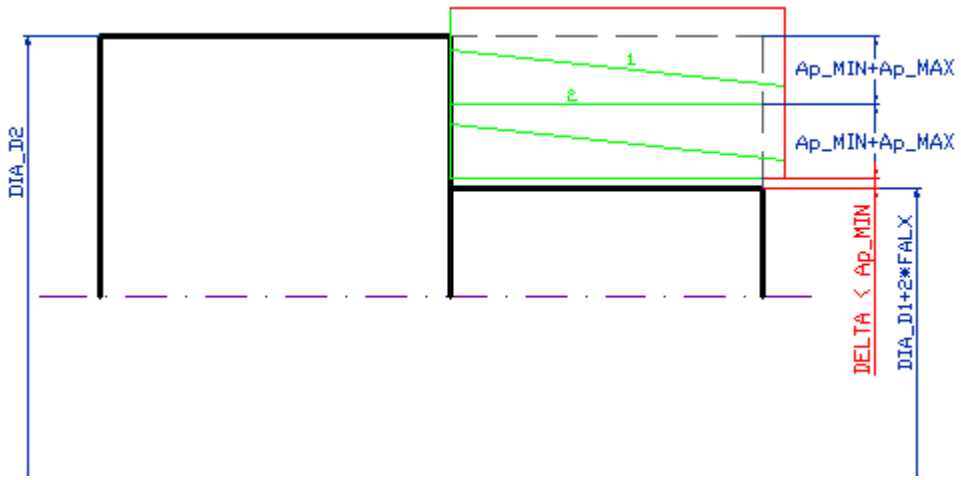
Obr. 3.2 Varianta 1 – úběr X je menší než Ap_MIN.

Druhá varianta – DELTA = 0, viz obr. 3.3. Využije se pouze BLOK_1. BLOK_2 se přeskočí za pomoci programového skoku GOTOF.



Obr. 3.3 Varianta 2 – DELTA = 0.

Třetí varianta – DELTA < Ap_MIN, viz obr. 3.4. Smyčka dráhy s proměnnou hloubkou záběru po druhé neproběhne, protože poslední záběr by byl menší než Ap_MIN. Na místo toho program skočí na BLOK_2, kde je tato situace vyřešena na dva záběry tak, aby vyhovovala technologickým požadavkům na Ap_MIN a Ap_MAX.



Obr. 3.4 Varianta 3 – DELTA < Ap_MIN.

Čtvrtá varianta – DELTA > Ap_MIN, viz obr. 3.5. V tomto případě bude zbylá část obrobena na jeden záběr jestliže DELTA>Ap_MIN a zároveň DELTA<=Ap_MAX nebo na dva záběry jestliže DELTA>Ap_MAX.

Obr. 3.5 Varianta 4 – DELTA > Ap_MIN.

3.2 Grafická podpora pro cyklus – CYCLE2000

Pro ulehčení práce obsluhy NC stroje při zadávání cyklu (CYCLE2000) byla vytvořena grafická podpora. V souboru **easyscreen.ini** se definuje prostředí, kde se bude nacházet softkey tlačítko, kterým uživatel grafickou podporu spustí. Dále je v souboru odkaz na soubor **hrub.com**, ve kterém je grafická podpora nadefinována, viz následující část programu easyscreen.ini:

```
StartFile15 = area := AreaProgramEdit, dialog := SlProgramEdit,
menu := SlStepStdTurnMenuHU, startfile := hrub.com
```

Definice masky – hrub.com

```
; ******** definice softkey ********

//S(START)

HS15=("HRUBOVANI")                    ; přiřazení názvu patnáctému horizontálnímu softkey
PRESS(HS15)                           ; akce pro stisk softkey

LM("maska1")                          ; stisknutím tlačítka s názvem HRUBOVANI se zobrazí grafická podpora (maska1)
END_PRESS

//END                                 ; konec definice softkey

; ******** definice masky ********

//M(maska1/"Muj cyklus CYCLE2000"/"obr_00.png") ; objeví se maska cyklu, v horní liště bude název – Můj cyklus CYCLE2000 a zobrazí se obrázek.

Def Var1=(R2///"DIA_D1","DIA_D1"//"obr_01.png"/"_HODN01"/370,50,150/430,50,100)
                                      ; definice lokální proměnné

...

VS7=("EXIT")                          ; přiřazení názvu sedmému vertikálnímu tl.

VS8=("PREVZIT")                       ; přiřazení názvu osmému vertikálnímu tl.

PRESS(VS7)
                                      ; stisknutím softkey VS7 se vrátí zpět do editace programu
EXIT

END_PRESS

PRESS(VS8)                            ; stisknutím softkey VS8 se vygeneruje (příkaz GC) VETA1
GC("VETA1")

EXIT

END_PRESS

OUTPUT(VETA1)                         ; výstup pro  VETA1

"CYCLE2000("var1","var2","var3","var4","var5","var6","var7","var8","var9","var10")"
END_OUTPUT

//END                                 ; konec definice masky cyklu
```

V souboru pro definici masky (hrub.com) se odkazuje na globální proměnné (např. _HODN01), které je nutné definovat v souboru UGUD.DEF. Tento soubor se nachází v příloze 5.

Veškeré obrázky od uživatele jsou uloženy v adresáři: Systém CF-Card/user/sinumerik/hmi/ico640.

Na obr. 3.6 je ukázka grafické podpory pro cyklus – CYCLE2000. Po vypsání hodnot a stisknutí tlačítka převzít, se do programu na místo, kde byl kurzor, vypíše věta CYCLE2000(……), která odkazuje na daný cyklus a přiděluje mu dané hodnoty.

Obr. 3.6 Grafická podpora pro cyklus – CYCLE2000.

3.3 Cyklus pro obrábění šestihranu (SESTIHRAN)

Šestihran je ve strojírenské výrobě velmi často vyskytující se geometrický tvar (např. šrouby, matice aj.). Ve výrobě může často vzniknout požadavek na frézování šestihranu, buď za účelem rychlého vytvoření nějakého tvarově neobvyklého šroubu, nebo na hřídeli, která se následně klíčem uvádí do konkrétní polohy.

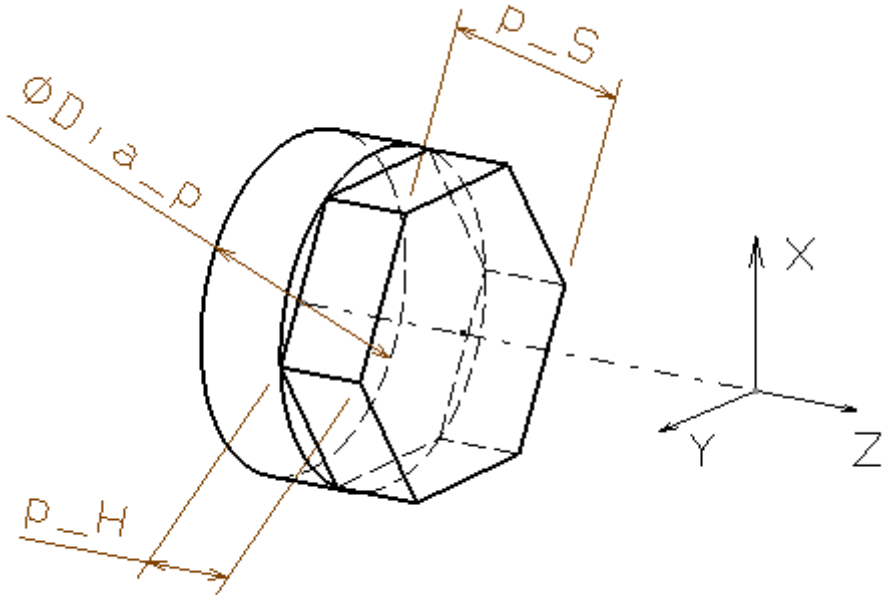
Úkolem je vytvořit takový cyklus, aby při zadání libovolné velikosti plochého/očkového klíče, byl vytvořen šestihran do požadované hloubky, za zvolených technologických podmínek. Takový cyklus může značně usnadnit a urychlit výrobu součásti, na níž se šestihran vyskytuje.

Opět je důležité, aby program fungoval při jakémkoliv zadání geometrických a technologických parametrů. V případě, že zadání bude chybné, program musí vypsat chybovou hlášku. V cyklu SESTIHRAN jsou zadávány parametry, viz tab. 3.2.

Tab. 3.2 Parametry, které jsou zadávány uživatelem cyklu.

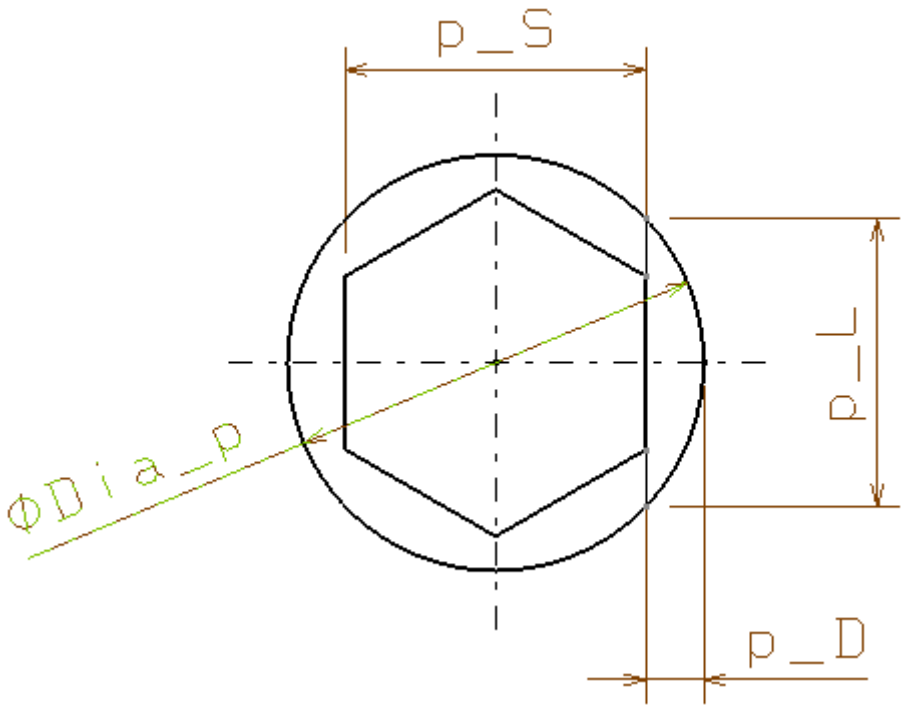
Symbol	Jednotka	Popis
p_H	[mm]	hloubka šestihranu
p_S	[mm]	číslo klíče (vzdálenost vodorovných ploch)
Dia_p	[mm]	průměr polotovaru
Dia_n	[mm]	průměr nástroje
Ap_RAD	[mm]	maximální radiální hloubka záběru
Ap_AX	[mm]	maximální axiální hloubka záběru
Lp	[mm]	délka přeběhu nástroje
FF1	[mm/min]	posuvová rychlost

Na obr. 3.7 je názorná ukázka šestihranu s geometrickými rozměry, které se v cyklu zadávají. Možná délka přeběhu nástroje Lp je v tomto případě nulová. Model součásti byl vytvořen v programu CATIA V5.



Obr. 3.7 Šestihran s rozměry, které se v cyklu zadávají.

K tomu, aby mohly být správně zadány souřadnice pohybu nástroje v ose Y, je zapotřebí znát délku tětivy L (v programu značeno p_L), viz obr. 3.8.



Obr. 3.8 Půdorys šestihranu a rozměry, potřebné k výpočtu tětivy.

Výpočet tětivy L je dle následujícího vztahu:

$$L = 2 \cdot \sqrt{D \cdot (Dia_p - D)} \tag{3.3}$$

kde D (p_D v programu) se vypočte dle vztahu:

$$D = \frac{1}{2}(Dia_n - S) \tag{3.4}$$

kde S (p_S v programu) je velikost klíče.

Stejně jako u předchozího cyklu, i zde se musela vyřešit otázka maximální hloubky ostří. V případě radiální hloubky záběru se hodnota Ap_RAD porovnává s hodnotou p_D a podle toho je určen počet zajetí v ose X. V případě axiální hloubky záběru se hodnota Ap_AX porovnává s hodnotou p_H a podle toho je určen počet zajetí v ose Z. Oba tyto výpočty si cyklus SESTIHRAN provádí sám. Blíže je toto řešení možno vidět v následující kapitole 4, kde je cyklus podrobně popsán.

4 EXPERIMENTÁLNÍ OVĚŘENÍ NC PROGRAMŮ

Pro experimentální ověření NC programů byl zvolen stroj SP280 (viz obr 4.1), který se nachází v prostorách laboratoře ústavu strojírenské technologie. Řídicím systémem stroje je Sinumerik 840Dsl. Jedná se o soustruh s dvěma vřeteny, třemi osami X, Y, Z a s možností natočení vřeten v ose Z. Na stroji lze provádět frézovací operace díky možnosti poháněných nástrojů.

Před experimentálním ověřením na samotném stroji byl NC program nejdříve testován v Sinutrainu, kde bylo možné odladit chyby a zkontrolovat správnost navržené technologie.

4.1 Zpracování a ověření NC programu cyklu CYCLE2000

Soubor cyklu CYCLE2000.SPF byl uložen do adresáře /NC Data/Cycles/User Cycles/. Pro testování byl cyklus vyvolán z hlavního programu s názvem MAIN.MPF.

Hlavní program - MAIN.MPF

```
N10 G54 G18 G71 G95 G90      ; nastavení základních G-funkcí
N20 T="pramet-hrub"          ; výměna nástroje – hrubovací nůž
N30 G96 S200 M4 M8           ; konst. řezná rychlost, směr otáčení, chlazení
N40 LIMS=2500                ; meze otáček obrobku na 2500 ot./min
N50 G0 X140 Z5               ; nájezd nástrojem před obrobek
N60 CYCLE2000(40,58,60,2,2.3,0.8,0,0,0.3,0)  ; volání cyklu
N70 G0 X140 Z5               ; přejezd do výměny nástroje
N80 M5 M9                    ; vypnutí otáček a chlazení
N90 M30                      ; konec programu
```

Cyklus – CYCLE2000

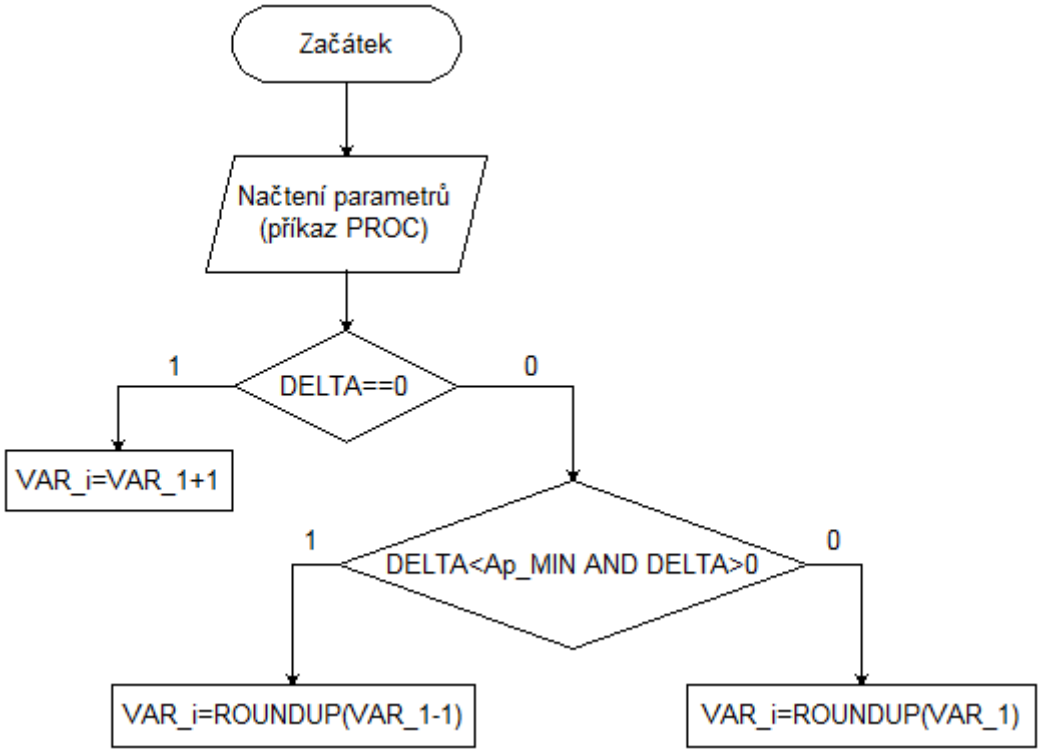
```
N10 PROC CYCLE2000 (REAL DIA_D1, REAL DIA_D2, REAL DELKA, REAL
L_n, REAL Ap_MAX, REAL Ap_MIN, REAL FALZ, REAL FALX, REAL FF1,
REAL DT) SAVE                                  ; předání parametrů

N20 DEF INT HRUB                               ; definice pomocného parametru

N30 DEF REAL XI, XII, VAR_1, VAR_i, DELTA      ; definice pomocných parametru
; ********** výpočty pomocných proměnných DELTA a VAR_1 **********

N40 DELTA=((DIA_D2-(DIA_D1+(2*FALX)))/2) MOD (Ap_MIN+Ap_MAX)

N50 VAR_1=((((DIA_D2-(DIA_D1+(2*FALX)))/2)/(Ap_MAX+Ap_MIN))-1)

; ********** podmínka uskutečnění cyklu **********

N60 IF ((DIA_D2-(DIA_D1+(2*FALX)))/2)<Ap_MIN

N70 DO SETAL(65001)                            ; chybová hláška
```

```
N80 M0                                          ; stop programu
N90 ENDIF                                       ; konec podmínky
;********** VAR_i **********
N100 IF DELTA==0                                ; pokud DELTA je celé číslo, provede se
N110 VAR_i=VAR_1+1                              ; následující výpočet (blok N110)
N120 ELSE
N130 IF (DELTA<Ap_MIN) AND (DELTA>0)
N140 VAR_i=ROUNDUP(VAR_1)-1                     ; zaokrouhlení hodnoty VAR_1 dolů
N150 ELSE                                       ; pomocí příkazu ROUNDUP a odečtení
                                                ; jedničky
N160 VAR_i=ROUNDUP(VAR_1)                       ; zaokrouhlení hodnoty VAR_1 nahoru
N170 ENDIF
N180 ENDIF
; **********  nastavení posuvu a najetí nástrojem před začátkem obrábění **********
N190 G95 F=(FF1)                                ; posuvová rychlost milimetrech na ot.
N200 G0 X=(DIA_D2+5) Z5                         ; najetí do polohy před začátkem
                                                ; obrábění
; ********** BLOK_1 **********
N210 FOR HRUB=1 TO VAR_i
...
; ********** BLOK_2 **********
N350 IF DELTA==0        ; jestliže DELTA je celé číslo, programový skok na KONEC
N360 GOTOF KONEC
N370 ELSE               ; jestliže DELTA celé číslo není, program se bude řídit podle
                        ; následujících variant v blocích N380, N450 a N570
N380 IF (DELTA>=Ap_MIN) AND (DELTA<=Ap_MAX)   ; Pouze jedno projetí
...
N450 IF DELTA>Ap_MAX                          ; Projet nadvakrát
...
N570 IF (DELTA<Ap_MIN) AND (DELTA>0)
```

```
...
N740 KONEC:          ; při zadání příkazu GOTOF KONEC, program skočí na tento řádek
N750 RET             ; Návrat do hlavního programu
```

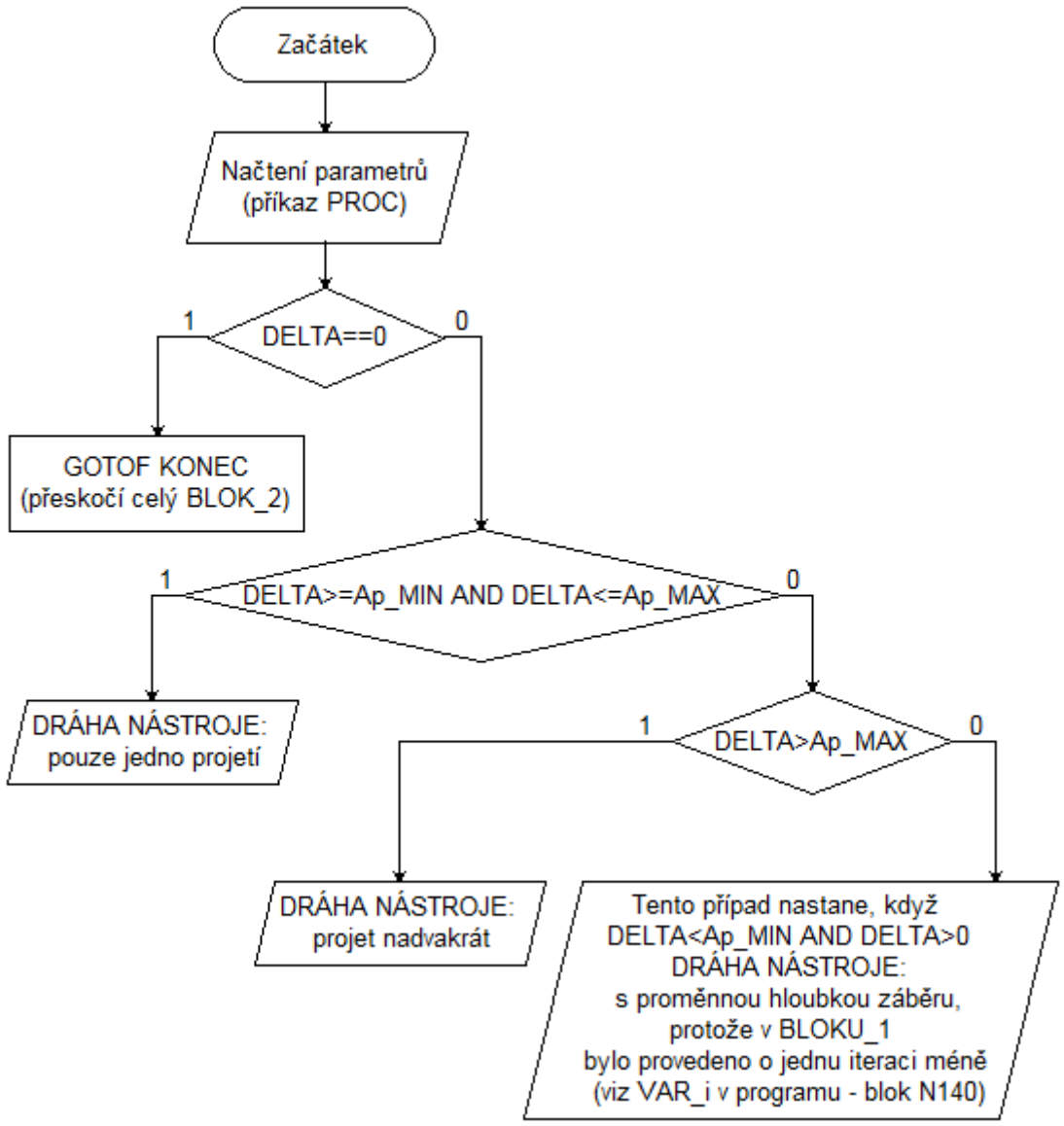
Celý NC program cyklu s názvem CYCLE2000 je v příloze 1.

V programu jsou na začátku definovány pomocné proměnné. Na základě proměnných VAR_i a DELTA jsou vykonány některé procesy, dle podmínek, které jsou v programu. VAR_i udává počet opakování smyčky v bloku 1 (BLOK_1). Tento počet opakování se mění v závislosti na Ap_MIN a Ap_MAX a také na podmínkách v blocích N100 až N180. Názorné zobrazení variant výpočtu parametru VAR_i na základě podmínek je v diagramu na obr. 4.1.



Obr. 4.1 Diagram podmínek výpočtu parametru VAR_i.

Jak už bylo zmíněno v kapitole 3, v druhém bloku (BLOK_2) je naprogramován zbytek pro obrobení, který už nelze obrobit proměnnou hloubkou záběru. Opět je zde několik variant kvůli libovolnému zadání hodnot Ap_MIN a Ap_MAX. Tyto varianty znázorňuje diagram na obr. 4.2.



Obr. 4.2 Vývojový diagram bloku 2 (BLOK_2).

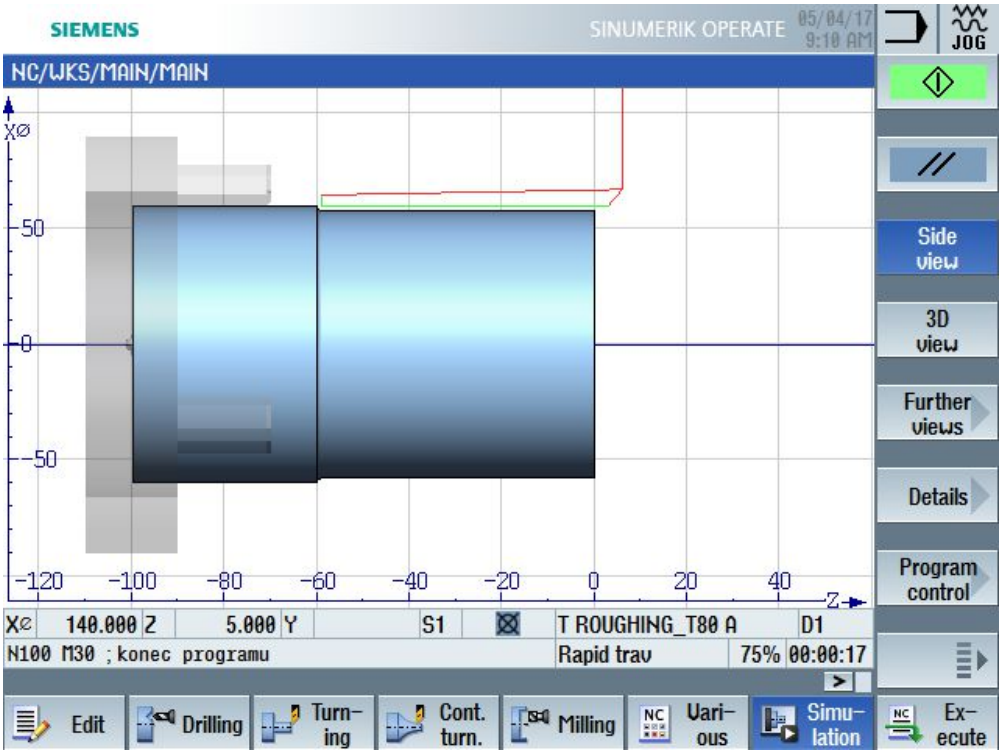
Na následujících obrázcích jsou znázorněny simulace dvou různých případů. První případ (DELTA>=Ap_MIN AND DELTA<=Ap_MAX) dle vývojového diagramu na obr. 4.2 bude dráha nástroje pouze na jedno projetí, viz obr. 4.3. Konkrétní zadané hodnoty jsou:

CYCLE2000(58, 60, 60, 2, 2.3, 0.8, 0, 0, 0.3, 0)

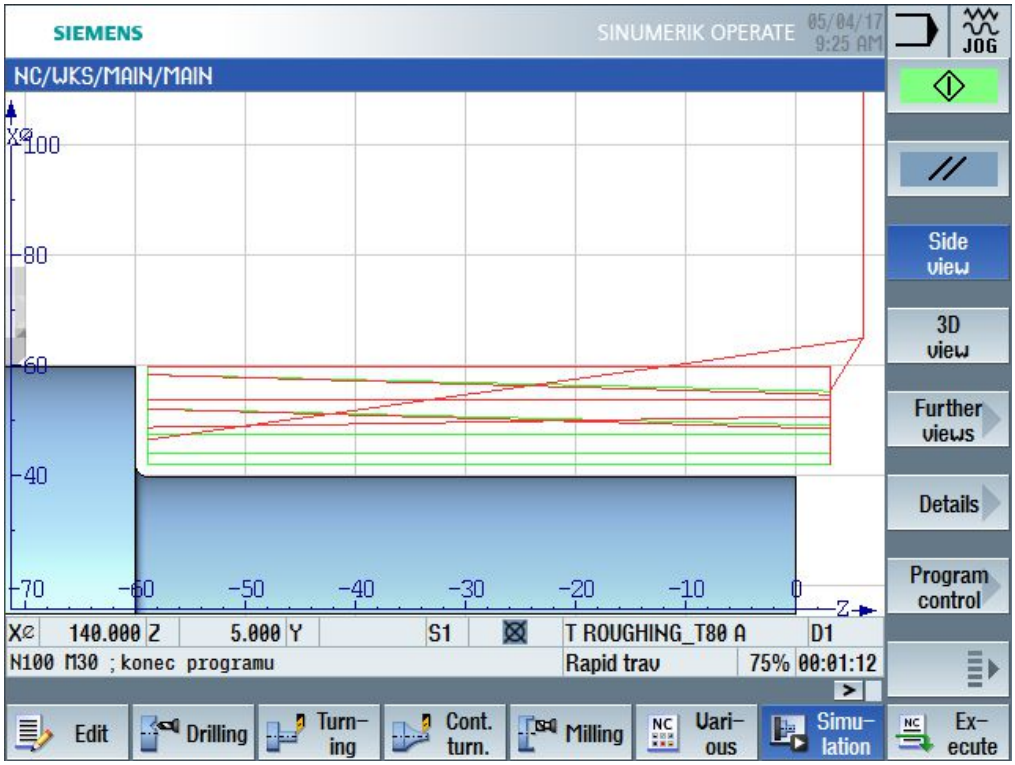
V druhém případě proběhly v BLOKU_1 dvě iterace (tzn. dvakrát proměnná hloubka záběru) a zbytek (DELTA>Ap_MAX) proběhl na dvě zajetí, viz obr. 4.4. Konkrétní zadané hodnoty jsou:

CYCLE2000(40, 58, 60, 2, 2.3, 0.8, 0, 0, 0.3, 0)

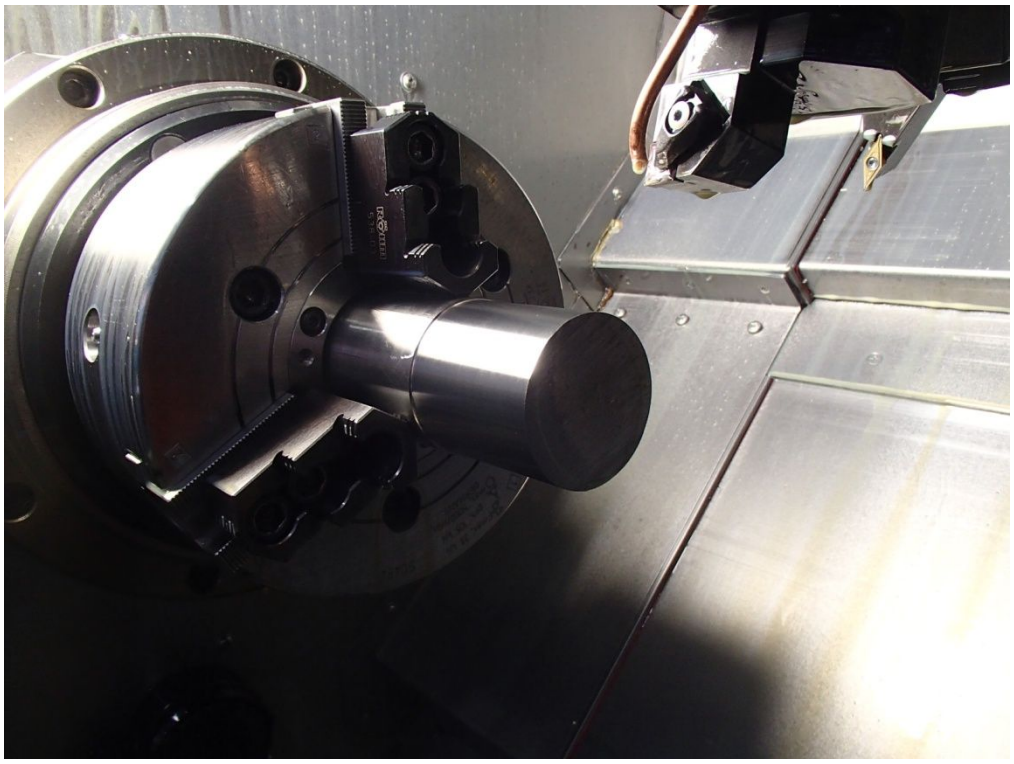
Tyto dvě náhodné varianty byly testovány i na stroji SP280. Materiálem obrobku byla korozivzdorná ocel, která byla k dispozici v laboratořích Ústavu Strojírenské Technologie. Samotný proces obrábění první varianty je na obr. 4.5 a výsledný obrobek druhé varianty je na obr. 4.6.



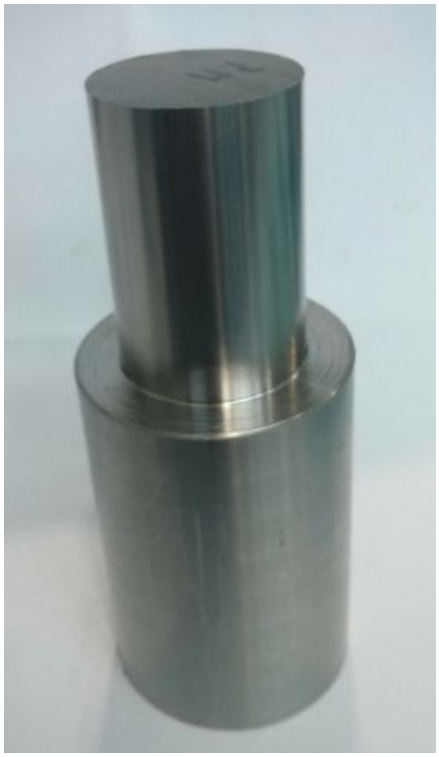
Obr. 4.3 Simulace NC programu cyklu s názvem CYCLE2000.



Obr. 4.4 Simulace NC programu cyklu s názvem CYCLE2000.



Obr. 4.5 Obrábění korozivzdorné oceli za pomoci hrubovacího cyklu CYCLE2000 – varianta 1.



Obr. 4.6 Hotový obrobek vyhotovený za pomoci hrubovacího cyklu CYCLE2000 – varianta 2.

4.2 Zpracování a ověření NC programu cyklu SESTIHRAN

Soubor cyklu SESTIHRAN.SPF byl opět uložen do adresáře /NC Data/Cycles/User Cycles/. Pro testování byl cyklus vyvolán z hlavního programu s názvem MAIN_SESTIHRAN.MPF. Program byl znovu sestaven na NC stroj SP280, na kterém je možnost poháněných nástrojů, tudíž je možné frézovat šestihran. Toto řešení je výhodné, protože součást, na které je zapotřebí soustružit šestihran je zpravidla rotační, takže se zvládnou obě operace soustružení i frézování na jednom stroji, na jedno upnutí nebo za použití protivřetena.

MAIN_SESTIHRAN.MPF

```
N10 G54 G71 G90                          ; nastavení základních G-funkcí
N20 T="FRÉZA 9HSS" M6                    ; volba nástroje
N30 G94 M3=3 S3=1200 M8                  ; lineární posuv, směr otáčení vřetena, otáčky, chlazení
N40 M14                                  ; vypnutí hlavního vřetena
N50 SESTIHRAN(6,30,36,9,1,0,200)         ; vyvolání cyklu
N60 M5 M9                                ; vypnutí otáček a chlazení
N70 M30                                  ; konec programu
```

SESTIHRAN.SPF

```
N10 PROC SESTIHRAN(REAL p_H, REAL p_S, REAL Dia_p, REAL Ap_RAD,
REAL Ap_AX, REAL Lp, REAL FF1)
                                         ; předání parametrů
;********* definice parametrů ********
N20 DEF REAL ZN, Ap_AX2, Ap_RAD2, ZN2, p_L, p_D     ; parametry jako reálná čísla
N30 DEF INT ZI, ZI2, ZI3, ZN3                       ; celočíselné hodnoty
; ******** výpočet parametrů ********
N40 p_D=(Dia_p-p_S)/2
N50 p_L=2*SQRT(p_D*(Dia_p-p_D))
N60 R1=(p_H+Lp) MOD Ap_AX                ; zbytek po dělení (MOD)
N70 IF R1==0                             ; jestliže R1 se rovná nule
N80 ZN=(p_H+Lp)/Ap_AX                    ; provede se tento výpočet
N90 ELSE                                 ; jestliže R1 se nerovná nule
N100 ZN=ROUNDUP((p_H+Lp)/Ap_AX)          ; provede se výpočet se zaokrouhlením
N110 ENDIF                               ; konec podmínky
N120 Ap_AX2=(p_H+Lp)/ZN                  ; úprava hodnoty axiální hloubky záběru
N130 R2=p_D MOD Ap_RAD
N140 IF R2==0
```

```
N150 ZN2=p_D/Ap_RAD
N160 ELSE
N170 ZN2=ROUNDUP(p_D/Ap_RAD)
N180 ENDIF
N190 Ap_RAD2=p_D/ZN2                ; úprava hodnoty radiální hloubky záběru
N200 ZN3=6                          ; jedná se o šestiúhelník
; ******** programové smyčky ********
N210 FOR ZI3=1 TO ZN3               ; smyčka proběhne tolikrát, kolik je stran
                                    (ZN3=6 protože chceme šestiúhelník)
N220 FOR ZI2=1 TO ZN2               ; počet iterací je závislý na radiální hloubce
                                    záběru Ap_RAD
N230 G90 G0 X=(Dia_p/2)-Ap_RAD2*ZI2 Y=(p_L/2)+1 Z2 F=FF1
                                    ; najetí nástrojem před obrobek
N240 FOR ZI=1 TO ZN                 ; počet iterací je závislý na axiální hloubce
                                    záběru Ap_AX
N250 G90 G1 Z=-ZI*Ap_AX2            ; zajetí nástrojem v ose Z
N260 G91 Y=-(p_L+2)                 ; frézování v ose Y
N270 G0 X1 Y=p_L+2 Z2               ; rychloposuv
N280 G1 X-1                         ; zpět v ose X do původní polohy
N290 ENDFOR                         ; konec programové smyčky (blok N210)
N300 ENDFOR                         ; konec programové smyčky (blok N220)
N310 G90 Z2                         ; přejezd nástrojem dva milimetry za
                                    obrobek
N320 G91 CA1=60                     ; otočení vřetenem o 60°
N330 ENDFOR                         ; konec programové smyčky (blok N240)
N340 G90 G0 X=(Dia_p/2)+5 Y=0 Z10   ; odjetí od obrobku před ukončením cyklu
N350 RET                            ; návrat do hlavního programu
```

Celkem byly v programu cyklu využity tři smyčky (příkaz FOR). První programová smyčka udává, kolik hran se bude soustružit, respektive kolikrát se obrobek má otočit, aby mohla být frézována další hrana. Tato smyčka je závislá na parametru ZN3, který je roven šesti. Bylo by tedy snadné udělat v případě potřeby z tohoto cyklu obecný cyklus na mnohohran. Pouze by se přidal tento parametr jako volitelný uživatelem. Druhá programová smyčka je závislá na Ap_RAD což vysvětluje následující příklad:

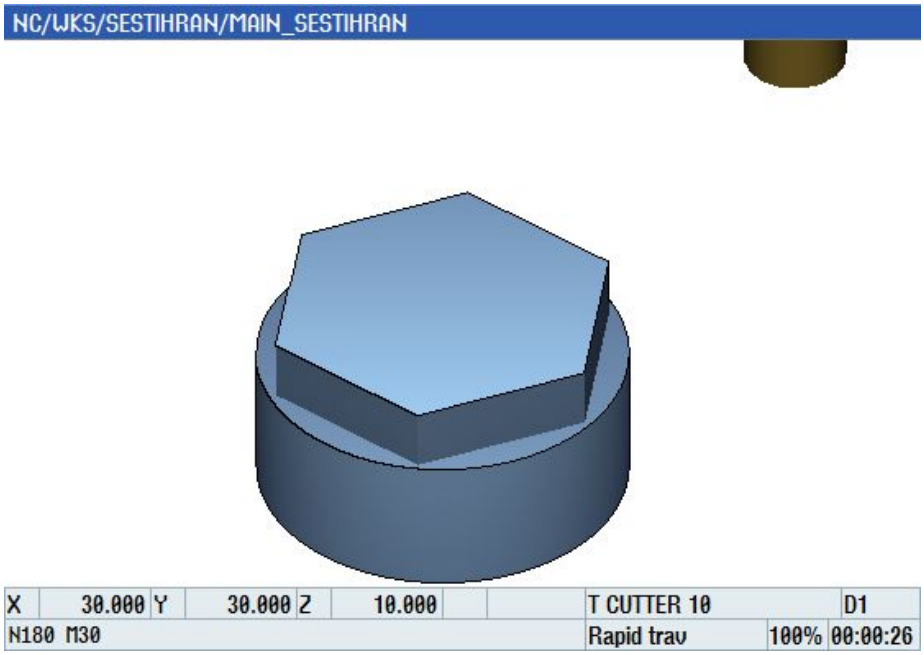
Jestliže Ap_RAD = 1 a je zapotřebí odebrat dva milimetry třísky, což udává v tomto případě parametr p_D, bude počet iterací smyčky roven dvěma, protože ZN2=p_D/Ap_RAD = 2/1 = 2.

V případě, že Ap_RAD = 1 a p_D = 2.5, bude dle výpočtu ZN2=ROUNDUP(p_D/Ap_RAD) = ROUNDUP(2.5/1) = 3. Hodnota radiální hloubky

záběru se přepočítá dle vztahu Ap_RAD2=p_D/ZN2=2.5/3=0.83333333. Tzn., počet iterací smyčky bude dle ZN2 rovno třem a radiální hloubka záběru Ap_RAD2, s kterou se dále počítá je rovna 0.83333333. Povoleno je v ŘS Sinumerik maximálně 8 desetinných míst, což je pro tyto účely více než dostatečné.

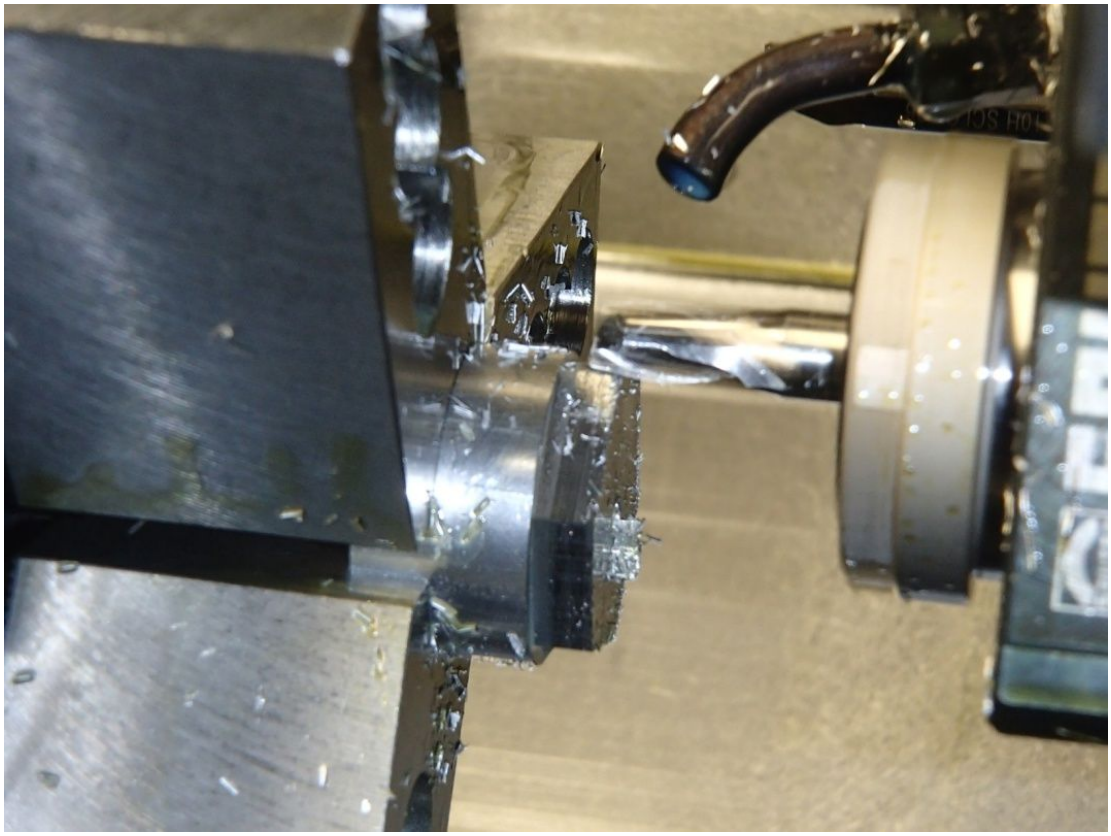
Pro třetí smyčku platí obdobná pravidla jako pro druhou s tím rozdílem, že je závislá na Ap_AX.

Simulaci cyklu SESTIHRAN je možno vidět na obr. 4.7. Na obr. 4.8 je výsledná součást frézovaná za pomoci cyklu SESTIHRAN a na obr. 4.9 je samotný proces frézování.



Obr. 4.7 Simulace NC programu cyklu SESTIHRAN.

Obr. 4.8 Hotový obrobek za pomoci frézovacího cyklu SESTIHRAN.



Obr. 4.9 Ukázka frézování šestihranu bez použití chlazení.

5 DISKUZE

5.1 Diskuze k praktické části

U samotného testování obou vytvořených programů na CNC stroji, nedošlo k žádným zásadním chybám. Díky simulaci v SinuTrainu byl program vždy předem odladěn na počítači a následně i na samotném stroji za snížené rychlosti posuvu.

5.1.1 Program cyklu – CYCLE2000

Bylo by zajímavé pokračovat v tvorbě tohoto cyklu pro hrubování těžkoobrobitelných ocelí a zdokonalit jej. Program byl vytvořen pouze pro jednoduchou aplikaci, kdy z libovolného průměru je potřeba soustružit na jiný průměr. Využití takového programu je značně omezené a bylo by velmi přínosné, kdyby cyklus uměl pracovat s libovolnou konturou, kterou by zadal uživatel, jako je tomu např. u cyklu ŘS Sinumerik – CYCLE95. Tvorba takového programu by si však žádala mnohem více času a hlubší rozbor po matematické i programátorské stránce.

5.1.2 Program cyklu – SESTIHRAN

Během tvorby tohoto cyklu bylo zjištěno, že stroj SP280, na kterém byl cyklus vyvíjen, obsahuje v části dílenského programování ShopTurn i cyklus na frézování mnohostranu. I tak bylo usouzeno, že tvorba cyklu má význam, protože tuto opci na dílenské programování stroj neobsahuje v základní verzi. Je to nástavba, za kterou si musí majitel stroje připlatit a ne každé firmě se vyplatí, tuto nástavbu mít.

Jak již bylo v kapitole 4 naznačeno a popsáno, bylo by následně snadné cyklus přepracovat na mnohohran, pokud by byl takový požadavek z výroby zadán.

ZÁVĚR

Tato práce měla za cíl mj. poukázat na to, že parametrické programování CNC strojů má obrovský potenciál pro automatizaci programování. Možnosti parametrického programování jsou takřka neomezené. Záleží pouze na odborné znalosti programátora a na jeho kreativitě. Dle výrobního programu firmy může programátor vytvořit řadu cyklů, které následně mohou ulehčit práci do budoucna. Nemusí se jednat pouze o přípravu cyklů. Parametrické programování je vhodné také pro komplexní řešení tvarově podobných součástí.

V práci je předveden způsob tvorby, využití a aplikace parametrického programování. V teoretické části byly předvedeny některé pokročilé funkce pro Sinumerik, které byly následně v praktické části využity. Byly naprogramovány dva obráběcí cykly, které mají ulehčit práci programátora a rapidně snížit čas na přípravu programu. Tyto cykly byly podrobně vysvětleny a popsány v rámci třetí a čtvrté kapitoly. U tvorby cyklů je důležité, aby byly postaveny na širokých základech, byly stabilní a flexibilní. Funkčnost a stabilitu cyklů dokazují experimentální zkoušky, které probíhaly na CNC stroji SP280 a jsou popsány v kapitole čtyři.

K jednomu cyklu byla vytvořena i grafická podpora, která má za cíl usnadnit používání cyklu. Postup její tvorby byl v práci také popsán.

Ve výrobní praxi se dnes stále více využívají CAD/CAM systémy, ale ne vždy je jejich aplikace vhodná. Např. parametrický program SESTIHRAN, který byl vytvořen v této práci, usnadňuje tvorbu programu na výrobu libovolného šestihranu, a to pouhým definováním sedmi parametrů. Následná příprava výroby, za pomoci takového programu, je i ve srovnání s dnešními vysoce produktivními CAD/CAM systémy efektivnější. Výhodou je také to, že si programátor může sestavit program na výrobu součásti přesně podle vlastních potřeb nebo potřeb výroby. V případě druhého vytvořeného obráběcího cyklu (CYCLE2000) byl vytvořen program, ve kterém jsou dráhy nástroje naprogramovány tak, aby docházelo k rovnoměrnému opotřebení břitové destičky. Je tak využit poznatek o obrábění těžkoobrobitelných ocelí, kdy dochází k nestejnoměrnému opotřebení nástroje v důsledku deformačního zpevnění obrobené vrstvy. Tento poznatek je následně aplikován do tohoto cyklu.

Díky konkrétním příkladům a podrobně zpracovaným programům s vysvětlivkami, může tato práce sloužit jako učební pomůcka nebo návod pro vytvoření vlastních uživatelských cyklů pro generace dalších studentů a pro ty, kteří si chtějí prohloubit znalosti v oblasti programování CNC strojů a přispět tak svou prací k vyšší efektivnosti programování a následně k vyšší efektivitě výroby.

SEZNAM POUŽITÝCH ZDROJŮ

1. **LYNCH, Mike.** *Parametric Programming for Computer Numerical Control Machine Tools and Touch Probes: CNC's Best-kept Secret.* Dearborn : Society of Manufacturing Engineers, 1997. p. 433. 0-87263-481-7.
2. **Siemens AG.** *SINUMERIK 840D sl / 828D, Pro pokročilé, Programovací příručka.* NÜRNBERG: Siemens AG, 2010. str. 822. Order no.: 6FC5398-2BP20-1UA0.
3. **MCMAHON Chris, BROWNE Jimmie.** *CADCAM: principles, practise and manufacturing management.* 2nd. Harlow : Prentice Hall, 1998. str. 665. 0-201-17819-2.
4. **KIEF, Hans B. a Helmut A. ROSCHIWAL.** *CNC handbook*. New York: McGraw-Hill, c2013. ISBN 978-0-07-179948-5.
5. **ŠTULPA, Miloslav.** *CNC: programování obráběcích strojů.* Praha: Grada, 2015. ISBN 978-80-247-5269-3.
6. **ASWORTH, David.** Podklady pro výuku předmětu ADMAN. Lyngby: DTU, 2014.
7. **Siemens AG.** *Tvorba zadávací masky uživatelského cyklu.*
8. **Siemens AG.** *SINUMERIK Operate, Sinutrain, Jednoduché soustružení se systémem ShopTurn*. NÜRNBERG: Siemens AG, 2011. str. 224. Order no.: 6FC5095-0AB80-1UP1.
9. **POWERMILL**. *Funkce programu* [online]. [vid. 2017-05-15]. Dostupné z: https://www.powermill.cz/funkce-programu#novinky
10. **Siemens AG.** *SINUMERIK 840D sl / 828D, Základy, Programovací příručka.* NÜRNBERG: Siemens AG, 2010. str. 588. Order no.: 6FC5398-1BP20-1UA0.
11. **GROOVER, Mikell P.** *Automation, Production Systems, and Computer-Integrated Manufacturing.* 3rd ed. Upper Saddle River : Prentice Hall, 2007. str. 815. 978-0132393218.
12. **S. K. Sinha.** *CNC Programming using Fanuc Custom Macro B.* US: McGraw-Hill Professional, 2010. ISBN: 9780071713320.
13. **Siemens AG.** *SinuTrain for SINUMERIK Operate v4.7 ED.2-Basic* [software]. [vid. 2017-05-15]. Dostupné z: https://www.industry.siemens.com/topics/global/en/cnc4you/cnc_downloads/sinutrain_downloads/pages/download-preview-version-sinutrain-for-sinumerik-operate-v4-7-basic.aspx
14. **ZOUHAR, Jan.** *Podklady pro výuku předmětu HC1-Aplikace CAD/CAM v technologii I.* Brno: VUT, 2016.
15. **MCMAHON Chris, BROWNE Jimmie.** *CADCAM: principles, practise and manufacturing management.* 2nd. Harlow : Prentice Hall, 1998. str. 665. 0-201-17819-2.
16. **RESEARCHGATE.** *NURBS curves with identical control points and knot-vectors - only difference weight* [online]. [vid. 2017-05-15]. Dostupné z:

https://www.researchgate.net/figure/264887412_fig4_Fig-4-NURBS-curves-with-identical-control-points-and-knot-vectors-only-difference

17. **ŽÁRA, Jiří, Bedřich BENEŠ a Petr FELKEL.** *Moderní počítačová grafika.* Praha: Computer Press, 1998. ISBN 80-7226-049-9.
18. **Siemens AG.** *Řídicí systémy Sinumerik* [online]. [vid. 2017-05-15]. Dostupné z: http://www1.siemens.cz/ad/current/index.php?vw=0&ctxnh=3c76394997&ctxp=home&acceptcookies=true
19. **CNC-BOW.** *Řídicí systém Sinumerik 828D Basic T* [online]. [vid. 2017-05-15]. Dostupné z: http://cnc.bow.cz/informace/9-ridici-system-sinumerik-828d-basic-t/?return=/informace/
20. **Sinumerik 840D sl:** Brochure [online]. [vid. 2015-05-25]. Dostupné z: http://www.industry.usa.siemens.com/drives/us/en/cnc/systems-and-products/Documents/Brochure-SINUMERIK-840D-sl.pdf
21. **SINUMERIK Solution Line** [online]. [vid. 2015-05-10]. Dostupné také z: http://siemens.h4h.pl/_13%20A13%20SINUMERIK%20solution%20line.pdf
22. **MAREK, Jiří.** *Konstrukce CNC obráběcích strojů III.* Praha: MM publishing, 2014, 684 s. MM speciál. ISBN 978-80-260-6780-1.
23. **HEIDENHAIN.** *Produkty Heidenhain a jejich aplikace* [online]. 2015 [vid. 2017-05-15]. Dostupné z: http://www.heidenhain.cz/cs_CZ/produkty/
24. **MATTSON, Mike.** *CNC programming: principles and applications.* Albany: Delmar, 2002, 358 s. ISBN 0766818888.
25. **Siemens AG.** *SINUMERIK, Manual, Mold Making, 3 to 5-Axis Simultaneous Milling.* 09/2011. 2011. Order No. 6FC5095-0AB10-0BP2.
26. **FANUC.** *Flexible CNC systems and solutions* [online]. [vid. 2017-05-16]. Dostupné z: http://www.fanuc.eu/cz/cs
27. **KONSTRUKCE.** *Obrobitelnost nerezových ocelí* [online]. [vid. 2017-05-16]. Dostupné z: http://www.konstrukce.cz/clanek/obrobitelnost-nerezovych-oceli/
28. **MM Průmyslové Spektrum.** *Příručka pro technology: proces obrábění v nerezových ocelích* [online]. [vid. 2017-05-16]. Dostupné z: http://www.mmspektrum.com/clanek/prirucka-pro-technology-proces-obrabeni-v-nerezovych-ocelich-zaverecne-tipy.html
29. **PÍŠKA, Miroslav**. *Nerovnoměrné opotřebení břitů při soustružení korozivzdorných ocelí* [konzultace]. 10.4.2017.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

Zkratka	**Popis**
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CNC	Computer Numerical Control
HMI	Rozhraní mezi uživatelem a strojem Human-Machine Interface
HW	Hardware
NC	Numerical Control
NCU	Numerical Control Unit
NURBS	Non-uniform rational B-Spline křivky a plochy
PCU	Programmable central unit (počítač)

Symbol	**Jednotka**	**Popis**
Ap_AX	[mm]	maximální axiální hloubka záběru
Ap_MAX	[mm]	maximální hloubka záběru
Ap_MIN	[mm]	minimální hloubka záběru
Ap_RAD	[mm]	maximální radiální hloubka záběru
DELKA	[mm]	délka obráběné plochy
DIA_D1	[mm]	požadovaný průměr
DIA_D2	[mm]	průměr polotovaru
Dia_n	[mm]	průměr nástroje
Dia_p	[mm]	průměr polotovaru
DT	[s]	časová prodleva za účelem odlomení třísky
FALX	[mm]	přídavek v ose X
FALZ	[mm]	přídavek v ose Z
FF1	[mm] [mm/min]	posuvová rychlost (posuv na otáčku nebo za minutu)
L	[mm]	délka tětivy
L_n	[mm]	délka náběhu nástroje
Lp	[mm]	délka přeběhu nástroje
p_H	[mm]	hloubka šestihranu
p_S	[mm]	číslo klíče (vzdálenost vodorovných ploch)
VAR_1	[-]	pomocná proměnná k výpočtu počtu iterací smyčky

SEZNAM PŘÍLOH

PŘÍLOHA 1

```
N10 PROC CYCLE2000 (REAL DIA_D1, REAL DIA_D2, REAL DELKA, REAL
L_n, REAL Ap_MAX, REAL Ap_MIN, REAL FALZ, REAL FALX, REAL FF1,
REAL DT) SAVE
N20 DEF INT HRUB
N30 DEF REAL XI, XII, VAR_1, VAR_i, DELTA
;****************DEFINICE PROMENNYCH*************************
N40 DELTA=((DIA_D2-(DIA_D1+(2*FALX)))/2) MOD (Ap_MIN+Ap_MAX)
N50 VAR_1=((((DIA_D2-(DIA_D1+(2*FALX)))/2)/(Ap_MAX+Ap_MIN))-1)
;****************PODMÍNKA USKUTEČNĚNÍ CYKLU******************
N60 IF ((DIA_D2-(DIA_D1+(2*FALX)))/2)<Ap_MIN
N70 DO SETAL(65001) ; chybová hláška, která hlásí že záběr je
menší než Ap_MIN nebo že DIA_D1>DIA_D2
N80 M0
N90 ENDIF
;****************VAR_i***************************************
N100 IF DELTA==0 ; Pokud VAR_1 je celé číslo
N110 VAR_i=(VAR_1+1)
N120 ELSE
N130 IF (DELTA<Ap_MIN) AND (DELTA>0)
N140 VAR_i=ROUNDUP(VAR_1)-1
N150 ELSE
N160 VAR_i=ROUNDUP(VAR_1)
N170 ENDIF
N180 ENDIF
;******************POSUV***************************************
N190 F=(FF1) ;nastavení rychlosti posuvu na otáčku
N200 G0 X=(DIA_D2+5) Z5 ; poloha před začátkem obrábění
;******************BLOK_1**************************************
N210 FOR HRUB=1 TO VAR_i
N220 XI=HRUB*Ap_MAX
N230 XII=(HRUB*Ap_MAX)+((HRUB-1)*Ap_MIN)
N240 G0 X=(DIA_D2-(2*XII)) Z=(L_n)
N250 G1 X=(DIA_D2-(2*XII))+2*((XI/HRUB)-Ap_MIN) Z=-(DELKA-
FALZ)
```

```
N260 G4 F=(DT)
N270 G1 X=IC(Ap_MIN)
N280 G0 Z=(L_n)
N290 G0 X=(DIA_D2-(2*XII))-(2*Ap_MIN)
N300 G1 Z=-(DELKA-FALZ)
N310 G4 F=(DT)
N320 G1 X=IC(Ap_MAX)
N330 G0 X=(DIA_D2-(2*XII))-(2*Ap_MIN)+1 Z=(L_n)
N340 ENDFOR
;**********************BLOK_2**********************************
N350 IF DELTA==0
N360 GOTOF KONEC
N370 ELSE
N380 IF (DELTA>=Ap_MIN) AND (DELTA<=Ap_MAX); projet
najedenkrát
N390 G0 X=(DIA_D1+2*FALX) Z=(L_n)
N400 G1 Z=-(DELKA-FALZ)
N410 G4 F=(DT)
N420 G1 X=IC(Ap_MAX)
N430 G0 X=(DIA_D2+5) Z5; KONECNA POLOHA
N440 ELSE
N450 IF DELTA>Ap_MAX ; projet nadvakrát
N460 G0 X=(DIA_D1+2*FALX)+(2*DELTA*0.4) Z=(L_n)
N470 G1 Z=-(DELKA-FALZ)
N480 G4 F=(DT)
N490 G1 X=IC(Ap_MAX)
N500 G0 X=IC(1) Z=(L_n)
N510 G0 X=(DIA_D1+2*FALX) Z=(L_n)
N520 G1 Z=-(DELKA-FALZ)
N530 G4 F=(DT)
N540 G1 X=IC(Ap_MAX)
N550 G0 X=(DIA_D2+5) Z5 ; KONECNA POLOHA
N560 ELSE
N570 IF (DELTA<Ap_MIN) AND (DELTA>0)
```

```
N580 G0 X=(DIA_D1+2*FALX)+(2*(DELTA+Ap_MIN)) Z=(L_n)
N590 G1 X=(DIA_D1+2*FALX)+(2*(DELTA+Ap_MIN))+(2*(Ap_MAX-
Ap_MIN-DELTA)) Z=-(DELKA-FALZ) ;(-DELTA)-aby nepřesahovalo
Ap_MAX
N600 G4 F=(DT)
N610 G1 X=IC(Ap_MAX)
N620 G0 Z=(L_n)
N630 G0 X=DIA_D1+2*FALX
N640 G1 Z=-(DELKA-FALZ)
N650 G4 F=(DT)
N660 G1 X=IC(Ap_MAX)
N670 G0 X=(DIA_D2+5) Z5;KONECNA POLOHA
N680 ELSE
N690 GOTOF KONEC
N700 ENDIF
N710 ENDIF
N720 ENDIF
N730 ENDIF
N740 KONEC:
N750 RET
```

PŘÍLOHA 2

```
N10 PROC SESTIHRAN(REAL p_H, REAL p_S, REAL Dia_p, REAL
Ap_RAD, REAL Ap_AX, REAL Lp, REAL FF1)
;*************definice parametrů**************
N20 DEF REAL ZN, Ap_AX2, Ap_RAD2, ZN2, p_L, p_D
N30 DEF INT ZI, ZI2, ZI3, ZN3
;*************výpočet parametrů**************
N40 p_D=(Dia_p-p_S)/2
N50 p_L=2*SQRT(p_D*(Dia_p-p_D))
N60 R1=(p_H+Lp) MOD Ap_AX
N70 IF R1==0
N80 ZN=(p_H+Lp)/Ap_AX
N90 ELSE
N100 ZN=ROUNDUP((p_H+Lp)/Ap_AX)
N110 ENDIF
N120 Ap_AX2=(p_H+Lp)/ZN
N130 R2=p_D MOD Ap_RAD
N140 IF R2==0
N150 ZN2=p_D/Ap_RAD
N160 ELSE
N170 ZN2=ROUNDUP(p_D/Ap_RAD)
N180 ENDIF
N190 Ap_RAD2=p_D/ZN2
N200 ZN3=6
;************* programové smyčky**************
N210 FOR ZI3=1 TO ZN3
N220 MSG("ZI3="<<ZI3)
N230 FOR ZI2=1 TO ZN2
N240 G90 G0 X=(Dia_p/2)-Ap_RAD2*ZI2 Y=(p_L/2)+1 Z2 F=FF1
N250 FOR ZI=1 TO ZN
N260 G90 G1 Z=-ZI*Ap_AX2
N270 G91 Y=-(p_L+2)
N280 G0 X1 Y=p_L+2 Z2
N290 G1 X-1
```

```
N300 ENDFOR
N310 ENDFOR
N320 G90 Z2
N330 G91 CA1=60
N340 ENDFOR
N350 G90 G0 X=(Dia_p) Y=p_L Z10
N360 RET
```

PŘÍLOHA 3

```
StartFile15 = area := AreaProgramEdit, dialog :=
SlProgramEdit, menu := SlStepStdTurnMenuHU, startfile :=
hrub.com
```

PŘÍLOHA 4

```
;definice softkey
//S(START)
HS15=("HRUBOVANI")


PRESS(HS15)
LM("maska1") ;vyvolání masky
END_PRESS
//END


//M(maska1/"Muj cyklus CYCLE2000"/"obr_00.png")


Def
Var1=(R2///"DIA_D1","DIA_D1"//"obr_01.png"/"_HODN01"/370,50,15
0/430,50,100)


Def
Var2=(R2///"DIA_D2","DIA_D2"//"obr_02.png"/"_HODN02"/370,70,15
0/430,70,100)


Def
Var3=(R2///"DELKA","DELKA"//"obr_03.png"/"_HODN03"/370,90,150/
430,90,100)


Def Var4=(R2///"Ln-
NABEH","Ln"//"obr_04.png"/"_HODN04"/370,110,150/430,110,100)


Def
Var5=(R2///"Ap_MAX","Ap_MAX"//"obr_05.png"/"_HODN05"/370,130,1
50/430,130,100)
```

```
Def
Var6=(R2///"Ap_MIN","Ap_MIN"//"obr_06.png"/"_HODN06"/370,150,1
50/430,150,100)


Def Var7=(R2///"FALZ-PRIDAVEK V OSE
Z","FALZ"//"obr_07.png"/"_HODN07"/370,170,150/430,170,100)


Def Var8=(R2///"FALX-PRIDAVEK V OSE
X","FALX"//"obr_08.png"/"_HODN08"/370,190,150/430,190,100)


Def Var9=(R2///"FF1-
POSUV","FF1"//"obr_09.png"/"_HODN09"/370,210,150/430,210,100)


Def Var10=(R2///"DT-Casova prodleva za ucelem odlomeni
trisky","DT"//"obr_09.png"/"_HODN10"/370,230,150/430,230,100)


;definice softkey

VS7=("EXIT")

VS8=("PREVZIT")


PRESS(VS7)

EXIT

END_PRESS


PRESS(VS8)

GC("VETA1") ;příkaz GC - zkr. GenerateCode spouští metodu
generování NC-věty, jejíž obsah je dán návěstím("VETA1") a
procedurou ("OUTPUT")

EXIT

END_PRESS


OUTPUT(VETA1)

"CYCLE2000("var1","var2","var3","var4","var5","var6","var7","v
ar8","var9","var10")"

END_OUTPUT


//END
```

PŘÍLOHA 5

```
N10 DEF NCK REAL _HODN01
N20 DEF NCK REAL _HODN02
N30 DEF NCK REAL _HODN03
N40 DEF NCK REAL _HODN04
N50 DEF NCK REAL _HODN05
N60 DEF NCK REAL _HODN06
N70 DEF NCK REAL _HODN07
N80 DEF NCK REAL _HODN08
N90 DEF NCK REAL _HODN09
N100 DEF NCK REAL _HODN10
N110 M30
```